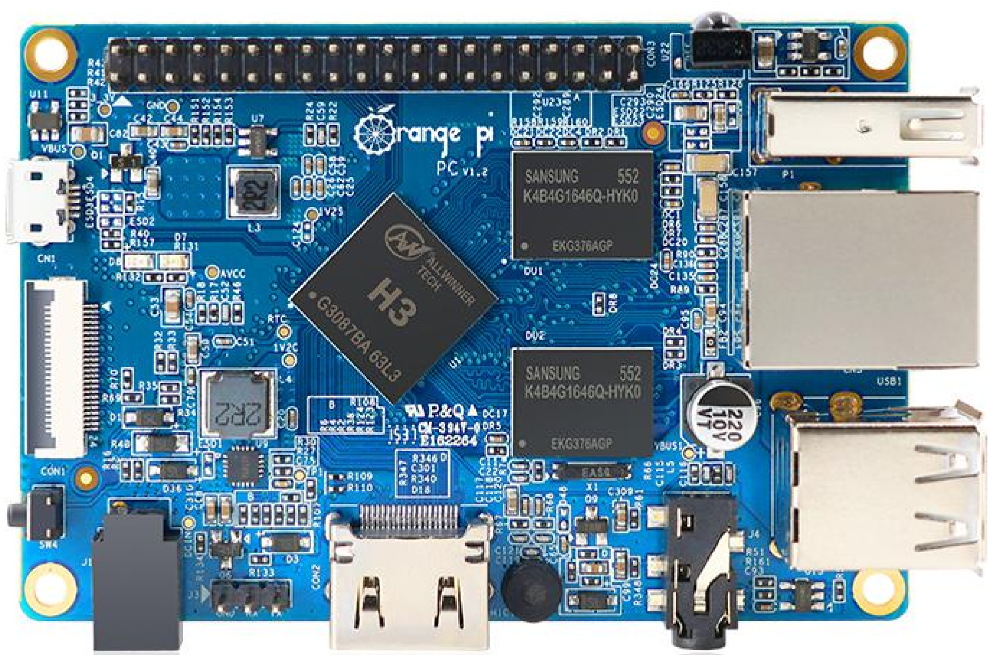
Orange Pi PC User Manual



Content

1. Basic features of Orange Pi PC

1.1. What's Orange Pi PC?

Orange Pi is an open-source single-board computer,a new generation of arm development board, it can run Android 4.4, Android 7.0, Ubuntu and Debian and other operating systems. Orange Pi PC uses the AllWinner H3 SoC, and has 1GB DDR3 memory

1.2. What can I do with Orange Pi PC?

We can use it to build:

- A computer
- A wireless network server
- Games
- Music player
- HD video player
- Speaker
- Android

Pretty much anything else, because Orange Pi Pc is open source

1.3. Who is for?

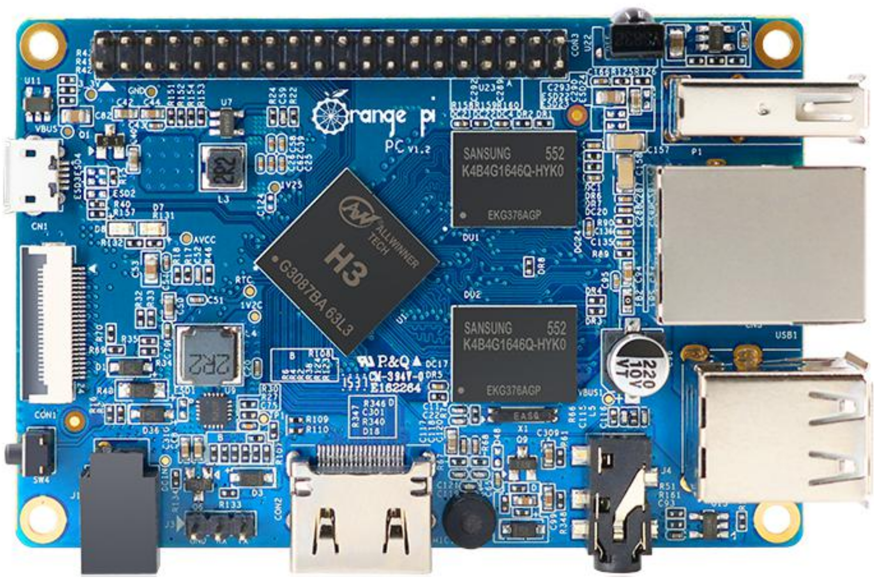
The Orange Pi development board is not only a consumer product, but also designed for anyone who wants to use technology to create and innovate. It is a very simple, interesting and practical tool, you can use it to create the world around you

1. 4. Orange Pi PC Hardware Specification

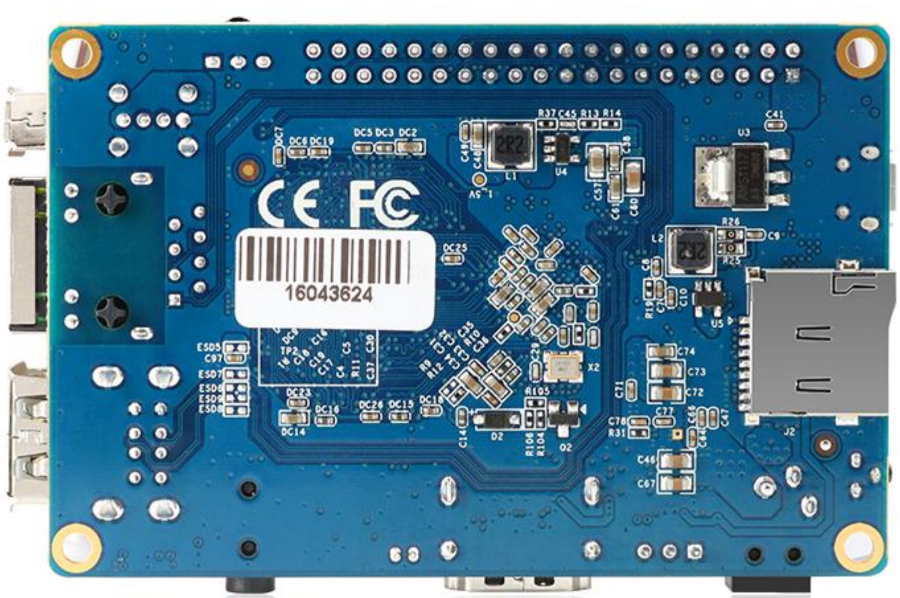
Hardware feature introduction	
CPU	Allwinner H3 ARM Cortex-A7 Quad Core
GPU	Mali400MP2 GPU @600MHz Supports OpenGL ES 2.0
Memory (SDRAM)	1GB DDR3 (shared with GPU）
Onboard Storage	TF card (Max. 32GB)
Onboard Network	10/100 Ethernet RJ45
Video Input	CSI interface camera
Audio Input	MIC
Video Outputs	HDMI, CVBS
Audio Output	3.5 mm audio port and HDMI
Power Source	DC input, MicroUSB (OTG) cannot be used as power input
USB 2.0 Ports	3*USB 2.0 HOST, 1*USB 2.0 OTG
Low-level peripherals	40 pin connector, compatible with Raspberry Pi
Debug serial port	UART-TX,UART-RX,GND
LED	Power led & Status led
Buttons	Power button（SW4）
IR receiver	Support IR remote control
Supported OS	Android,Ubuntu,Debian Image
Interface Definition	
Product Size	85mm×55mm
Weight	43g
range Pi™ is a trademark of the Shenzhen Xunlong Software CO., Limited	

1.5. Top and bottom views

Top View:

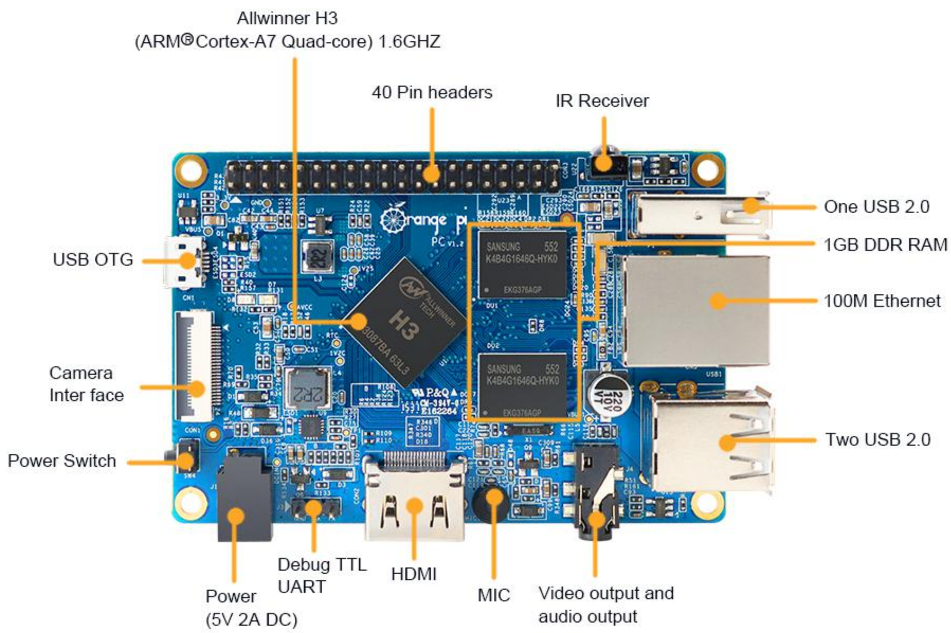


Bottom View:

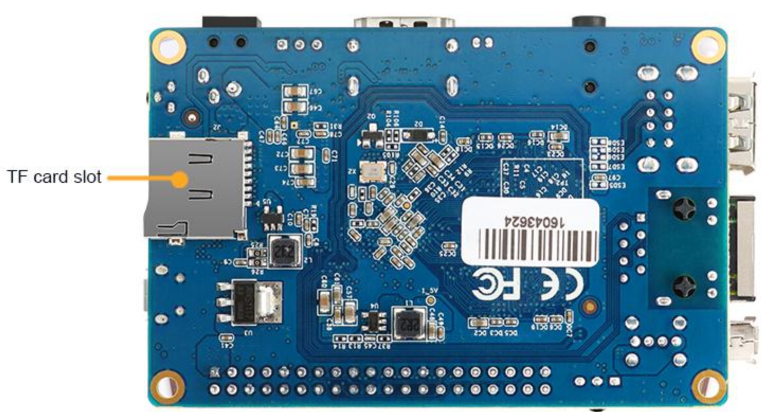


1.6. Interface Details

Top view



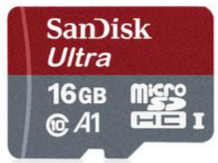
Bottm view



2. Introduction to the use of the development board

2.1. Prepare the necessary accessories

1) TF card, a high-speed card of class 10 or higher with a minimum capacity of 8GB, it is recommended to use SanDisk TF card, Orange Pi test is to use SanDisk TF card, other brands of TF card may have the problem of system failure



2) TF card reader, used to read and write TF card

3) Standard HDMI to HDMI cable, used to connect the development board to an HDMI monitor or TV for display

4) Power adapter, at least 5V/2A high-quality power adapter, note that the OTG interface of the development board cannot be used as a power input

5) USB interface mouse and keyboard, as long as it is a standard USB interface mouse and keyboard, the mouse and keyboard can be used to control the Orange Pi development board

6) IR remote control, mainly used to control Android system

7) 100M or Gigabit network cable, used to connect the development board to the Internet

8) GC2035 or OV5640 camera kit, which can be connected to the CSI interface of the development board to display video images

9) Micro USB interface data cable, when using the ADB debugging function of the Android system, you need to connect the development board to the computer through the Micro USB interface data cable

10) AV video cable, if you want to display video through the CVBS interface instead of the HDMI interface, then you need to connect the development board to the TV through the AV video cable

11) USB to TTL module and DuPont cable. When using the serial port debugging function, USB to TTL module and DuPont cable are required to connect the development board and the computer

12) A personal computer with Ubuntu and Windows operating systems

1	Ubuntu14.04 PC	Optional, used to compile Android source code
2	Ubuntu18.04 PC	Optional, used to compile Linux source code
3	Windows PC	Used to burn Android and Linux images

2.2. Download the image and related information of the development board

1) The download URL of the Chinese version is

```
http://www.orangepi.cn/downloadresourcescn/
```

2) The download URL of the English version is

```
http://www.orangepi.org/downloadresources/
```

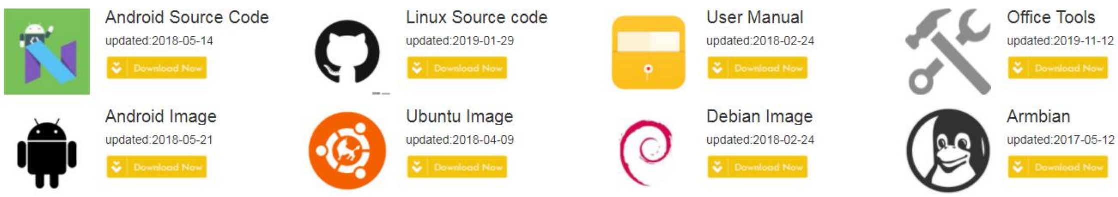
3) The information mainly contains

a. Android source code: saved on Baidu Cloud Disk and Google Cloud Disk
b. Linux source code: saved on github, the link address is

```
https://github.com/orangepi-xunlong
```

c. User manuals and schematic diagrams: chip-related data manuals will also be placed here
d. Official tools: mainly include the software that needs to be used during the use of the development board
e. Android image: saved on Baidu Cloud Disk and Google Cloud Disk
f. Ubuntu image: saved on Baidu Cloud Disk and Google Cloud Disk
g. Debian image: saved on Baidu Cloud Disk and Google Cloud Disk
h. Armbian image, a image developed by the Armbian community. If you encounter any problems during use, please report to the armbian forum first. The maintainer of the Armbian image and other people who use the Armbian image

will assist in solving various problems. This is also a fastest solution way to the problem.Orange Pi is not responsible for maintaining this image.



2. 3. Method of flashing Linux image to TF card based on Windows PC

1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

2) Then use the card reader to insert the TF card into the windows computer

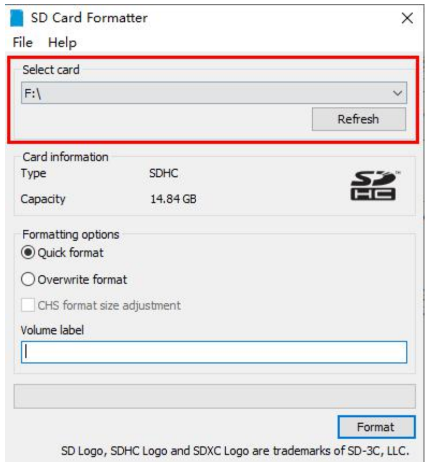
3) Then format the TF card

a. You can use the SD Card Formatter software to format the TF card, the download address is

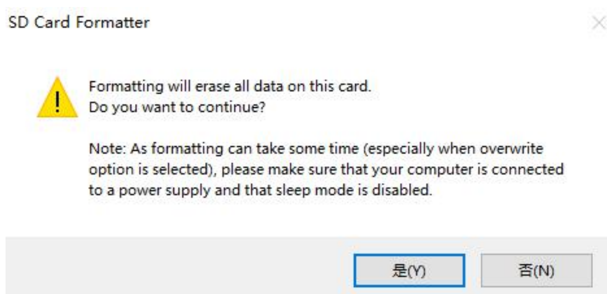
```
https://www.sdcard.org/downloads/formatter/eula_windows/SDCardFormatterv5_WinEN.zip
```

b. After downloading, you can directly unzip and install, and then open the software

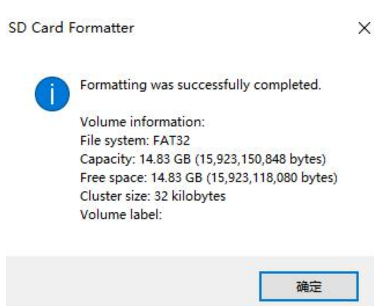
c. If the computer only has a TF card inserted, the TF card's drive letter will be displayed in the Select card column. If the computer has multiple USB storage devices inserted, you can select the drive letter corresponding to the TF card through the drop-down box



d. Then click Format, a warning box will pop up before formatting, and formatting will start after selecting "Yes (Y)"



e. After formatting the TF card, the message shown in the figure below will pop up, click OK



4) Download the Linux operating system image file compression package you want to burn from the Orange Pi data download page, and then use the decompression software to decompress it. In the decompressed file, the file ending with ".img" is the operating system image file. The size is generally above 1GB

5) Use Win32Diskimager to burn Linux image to TF card

a. The download page of Win32Diskimager is

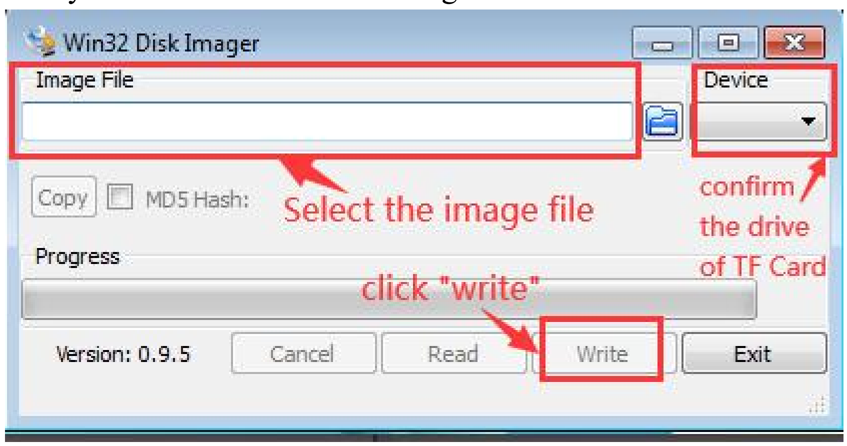
```
http://sourceforge.net/projects/win32diskimager/files/Archive/
```

b. Install directly after downloading, the interface of Win32Diskimager is shown below

a) First select the path of the image file

b) Then confirm that the drive letter of the TF card is consistent with the one displayed in the "Device" column

c) Finally click "write" to start burning



c. After the image is written, click the "Exit" button to exit, and then you can pull out the TF card and insert it into the development board to start

2. 4. Method of flashing Linux image to TF card based on Ubuntu PC

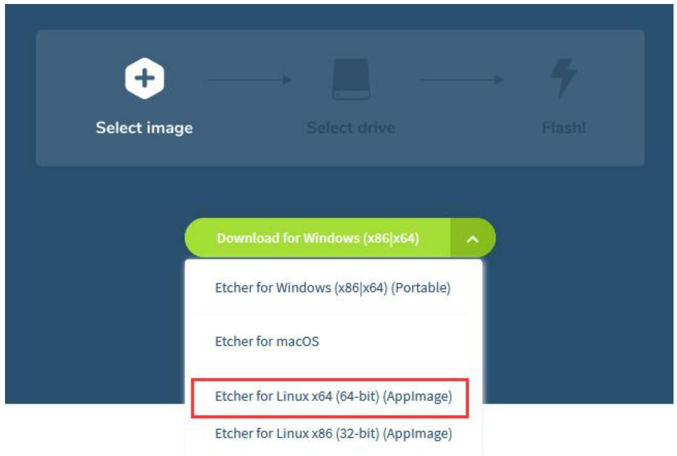
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

2) Then use a card reader to insert the TF card into the computer

3) Download balenaEtcher software, the download address is

```
https://www.balena.io/etcher/
```

4) After entering the balenaEtcher download page, please select the Linux version of the software through the drop-down box to download



5) After downloading, use unzip to decompress, the decompressed balenaEtcher-1.5.109-x64.AppImage is the software needed for burning

```
test@test:~$ unzip balena-etcher-electron-1.5.109-linux-x64.zip
Archive:   balena-etcher-electron-1.5.109-linux-x64.zip
  inflating: balenaEtcher-1.5.109-x64.AppImage
test@test:~$ ls
balenaEtcher-1.5.109-x64.AppImage   balena-etcher-electron-1.5.109-linux-x64.zip
```

6) Download the Linux operating system image file compression package you want to burn from the Orange Pi data download page, and then use the decompression software to decompress it. In the decompressed file, the file ending with ".img" is the operating system image file. The size is generally above 1GB

a. The decompression command of the compressed package at the end of 7z is as follows

```
test@test:~$ 7z x   image_filename.7z
```

b. The decompression command for the compressed package at the end of tar.gz is as follows

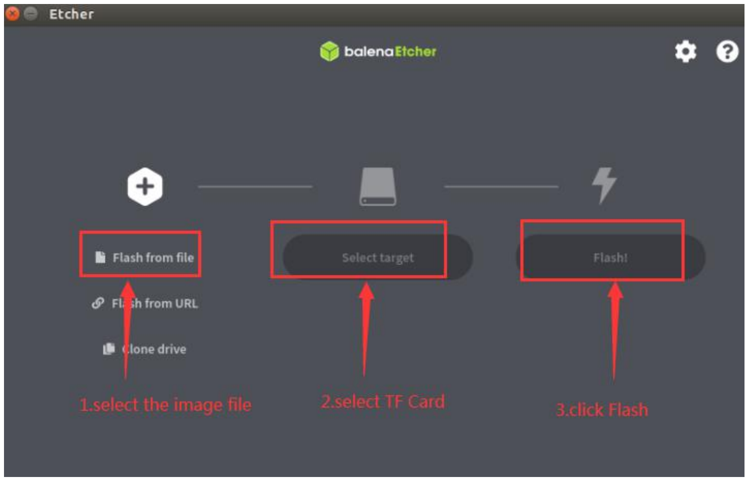
```
test@test:~$ tar -zxf   image_filename.tar.gz
```

7) Double-click balenaEtcher-1.5.109-x64.AppImage on the graphical interface of Ubuntu PC to open balenaEtcher. The opened interface is shown in the figure below

a. First select the path of the image file

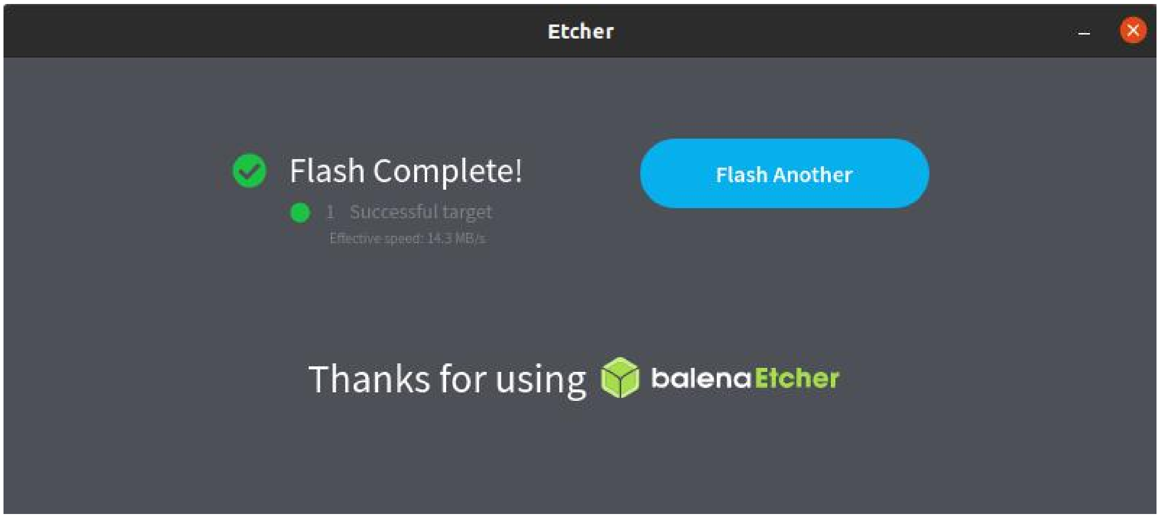
b. Then select the device number of the TF card

c. Finally click Flash to start burning



8) The writing speed and remaining time will be prompted during the burning process

9) After burning, the following interface will be displayed. At this time, you can unplug the TF card from the computer and insert it into the development board to start.



2. 5. Method of flashing Android firmware to TF card

Android image can only be burned to TF card using PhoenixCard software under Windows platform, but cannot be burned under Linux platform

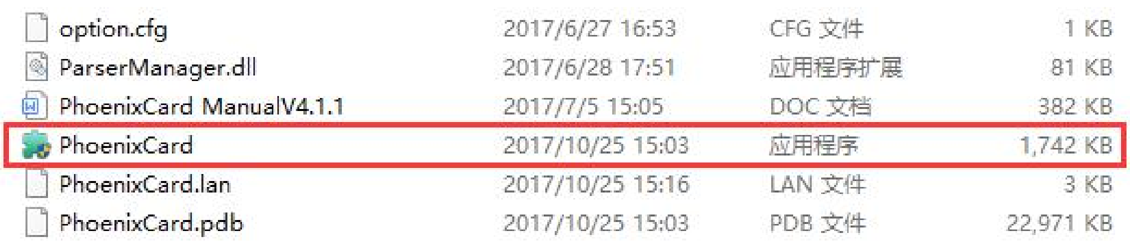
1) First, prepare a TF card with 8GB or larger capacity. The transmission speed of the TF card must be above class10. It is recommended to use a TF card of SanDisk and other brands

2) Then use a card reader to insert the TF card into the computer

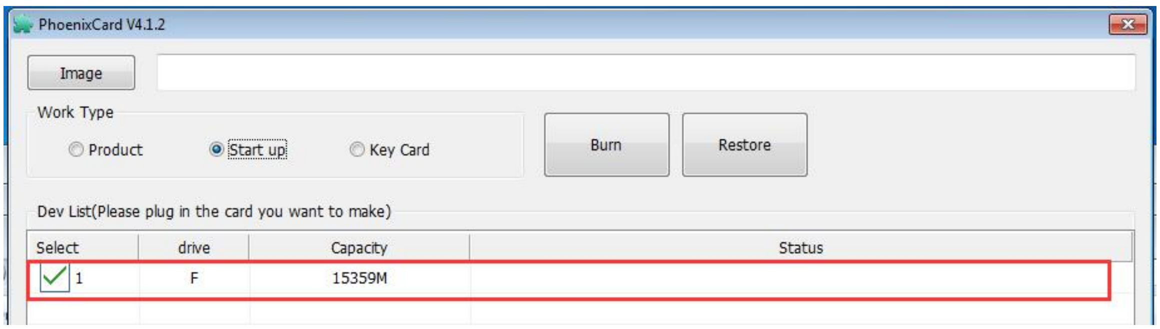
3) **Download Android 4.4 or Android 7.0 firmware and PhoenixCard burning tool from Orange Pi's data download page, please make sure that the phonenixCrad tool version is PhoenixCard v4.1.2**

4)Use the decompression software to decompress the compressed package of the downloaded Android firmware. In the decompressed file, the file ending with ".img" is the Android firmware

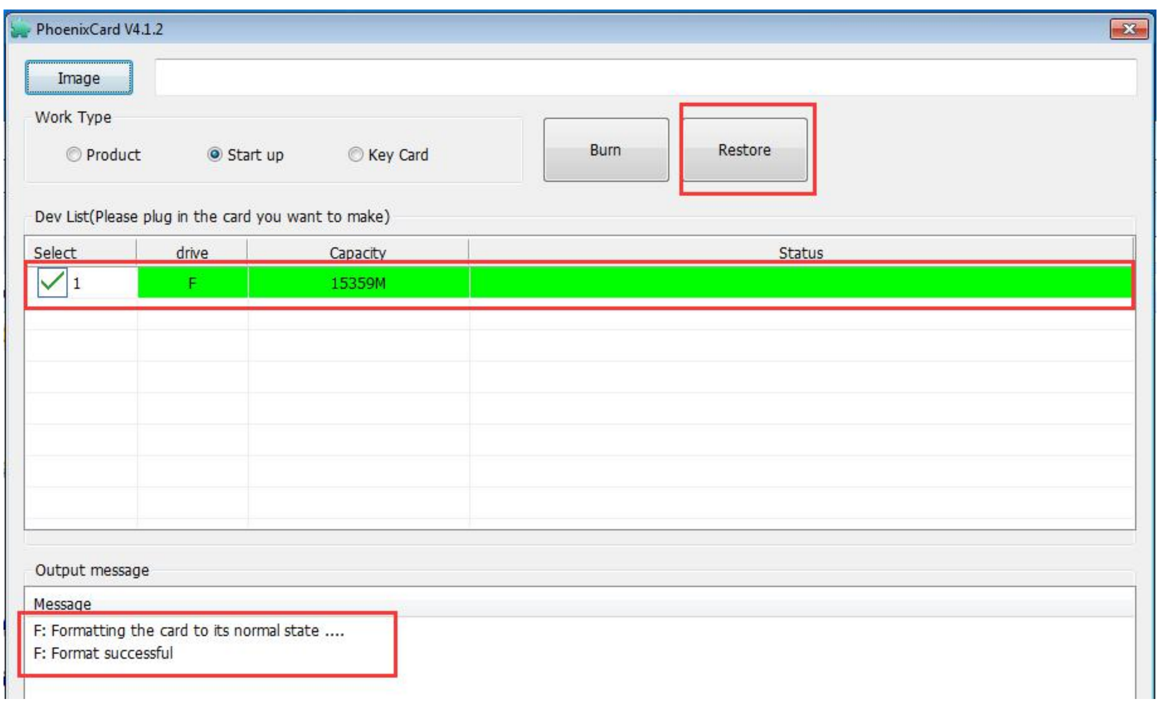
5) Use decompression software to decompress PhoenixCard v4.1.2.rar, this software does not need to be installed, you can find PhoenixCard in the decompressed folder and open it



6) After opening PhoenixCard, if the TF card is recognized normally, the drive letter and capacity of the TF card will be displayed in the middle list. Please make sure that the displayed drive letter is consistent with the drive letter of the TF card you want to burn. If it is not displayed You can try to unplug and plug the TF card



7) After confirming the drive letter, format the TF card first, click the **restore card** button in PhoenixCard, or use the aforementioned SD Card Formatter to format the TF card

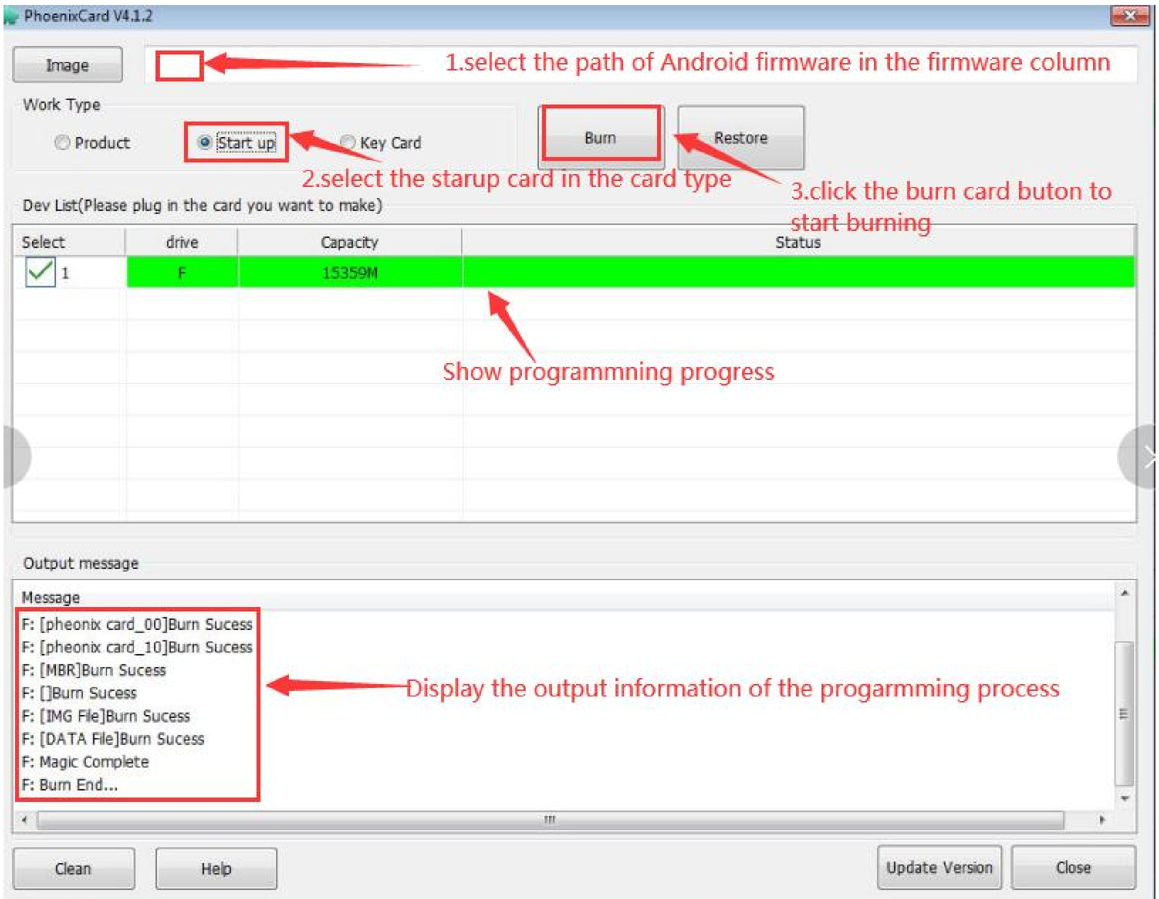


8) Then start to write the Android firmware to the TF card

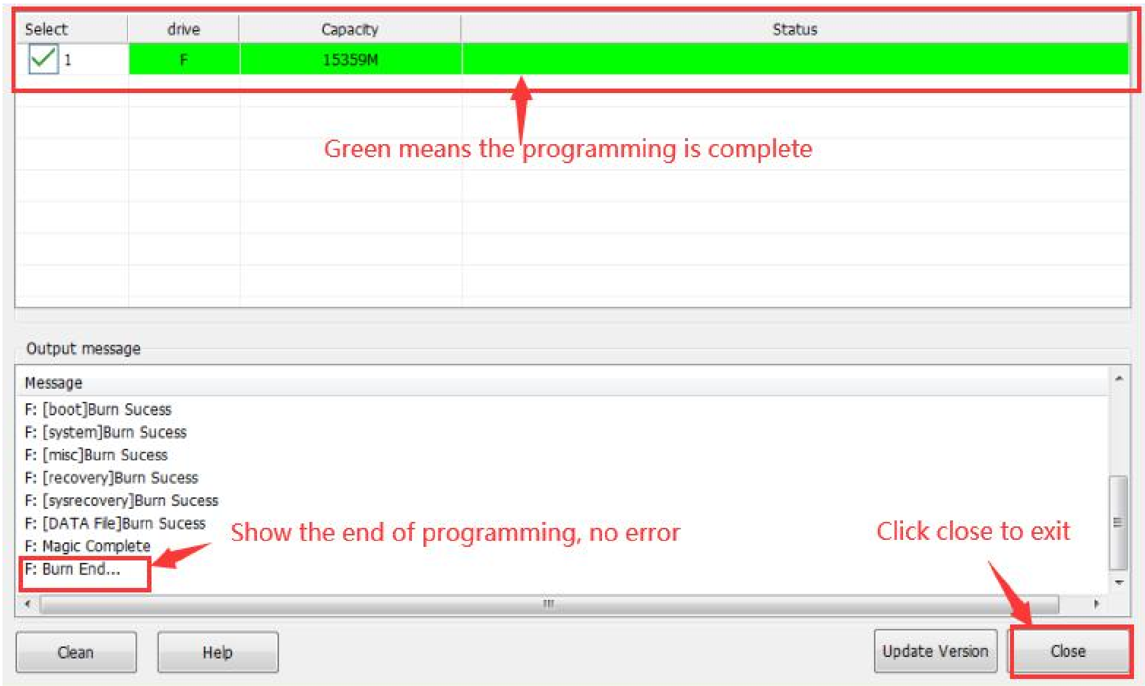
a. First select the path of Android firmware in the firmware column

b. Select the startup card in the card type

c. Then click the burn card button to start burning



9) After burning, the PhoenixCard will be displayed as shown in the figure below. At this time, click the close button to exit PhoenixCard, and then you can unplug the TF card from the computer and insert it into the development board to start.



2. 6. Start the Orange Pi development board

1) Insert the TF card with the burned image into the TF card slot of the Orange Pi development board. If there is a system in the eMMC, you can directly start the system in the eMMC without inserting the TF card.

2) The development board has an HDMI interface, you can connect the development board to a TV or other HDMI monitors through an HDMI cable

3) Connect the USB mouse and keyboard to control the Orange Pi development board

4) The development board has an Ethernet port, which can be plugged into a network cable for Internet access

5) Connect a 5V and at least 2A power adapter (3A is also possible)

   a. **Remember not to plug in the 12V power adapter, if you plug in the 12V power adapter, it will burn the development board**

   b. **Many unstable phenomena during system power-on and startup are basically caused by power supply problems, so a reliable power adapter is very important**

6) Then turn on the switch of the power adapter, if everything is normal, the HDMI display will be able to see the startup screen of the system at this time

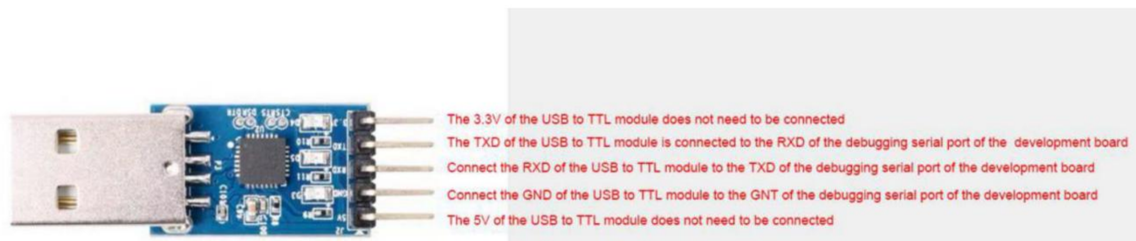
7) If you want to view the output information of the system through the debug serial port,

please use the serial cable to connect the development board to the computer. For the connection method of the serial port, please refer to the section on **the use of the debug serial port**

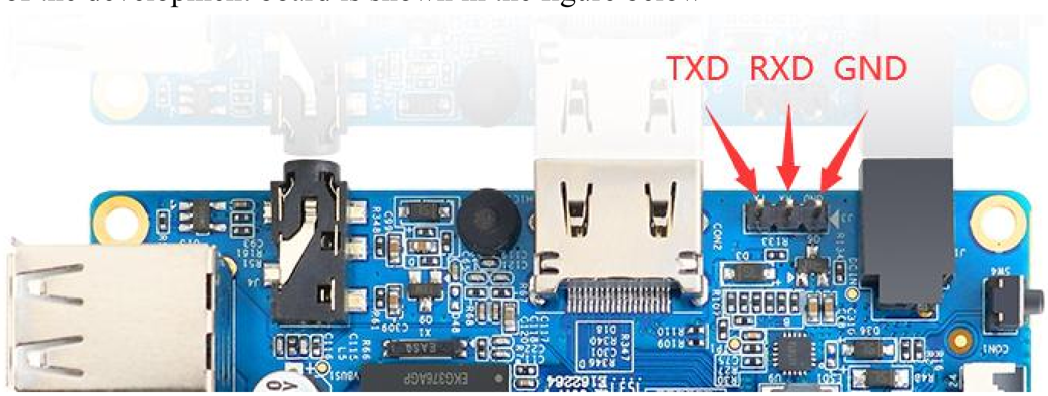
2. 7. How to use the debug serial port

2. 7. 1. Debug serial port connection instructions

1) First, you need to prepare a USB to TTL module. This module can be bought in Orange Pi stores. If there are other similar USB to TTL modules, you can also insert the USB end of the USB to TTL module into the USB port of the computer



2) The corresponding relationship between the debug serial port GND, TXD and RXD pins of the development board is shown in the figure below



3) The GND, TXD and RXD pins of the USB to TTL module need to be connected to the debug serial port of the development board through a DuPont cable

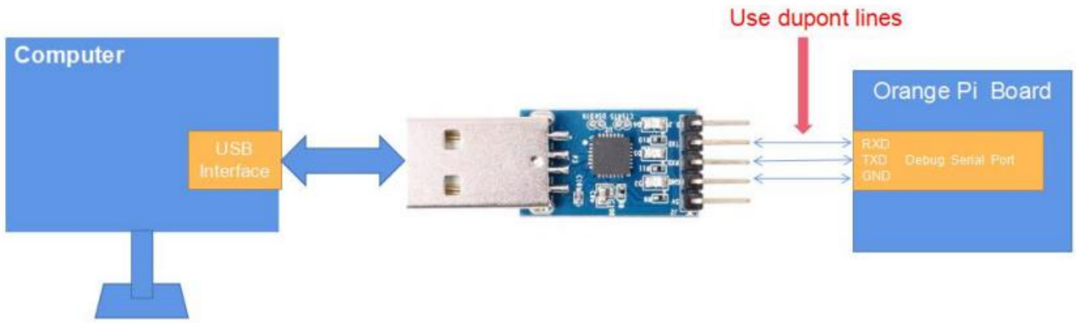
a. Connect the GND of the USB to TTL module to the GND of the development board

b. **Connect the RXD of the USB to TTL module to the TXD of the development board**

c. **Connect the TXD of the USB to TTL module to the RXD of the development**

board

4) The schematic diagram of connecting the USB to TTL module to the computer and the Orange Pi development board is shown below



Schematic diagram of connecting USB to TTL module to computer and Orange Pi development board

2. 7. 2. How to use the debug serial port on the Ubuntu platform

1) If the USB to TTL module is connected normally, you can see the corresponding device node name under /dev of Ubuntu PC, remember this node name, you will use it when setting up the serial port software later

```
test@test:~$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

2) There are many serial debugging tools that can be used under linux, such as putty, minicom, etc. The following shows how to use putty

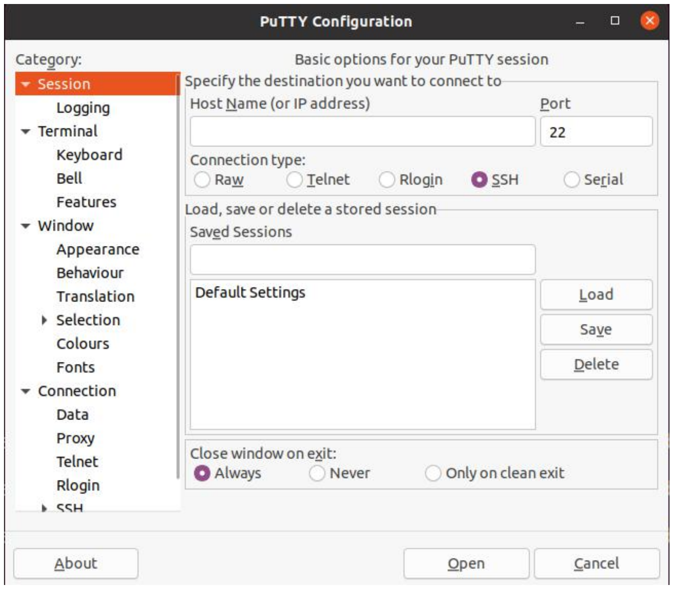
3) First install putty on the Ubuntu PC

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install putty
```

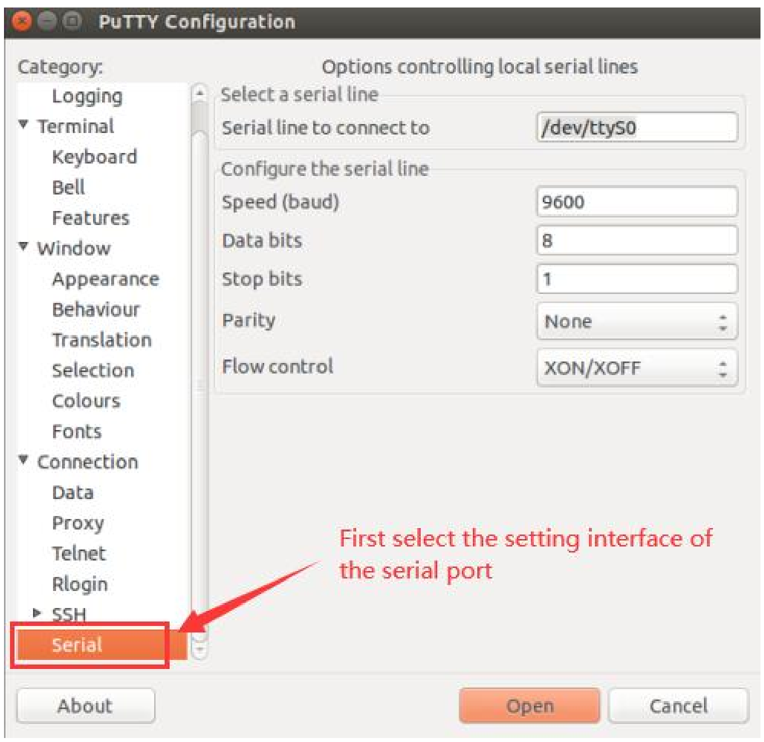
4) Then run putty, remember to add sudo permissions

```
test@test:~$ sudo putty
```

5) After executing the putty command, the following interface will pop up



6) First select the setting interface of the serial port

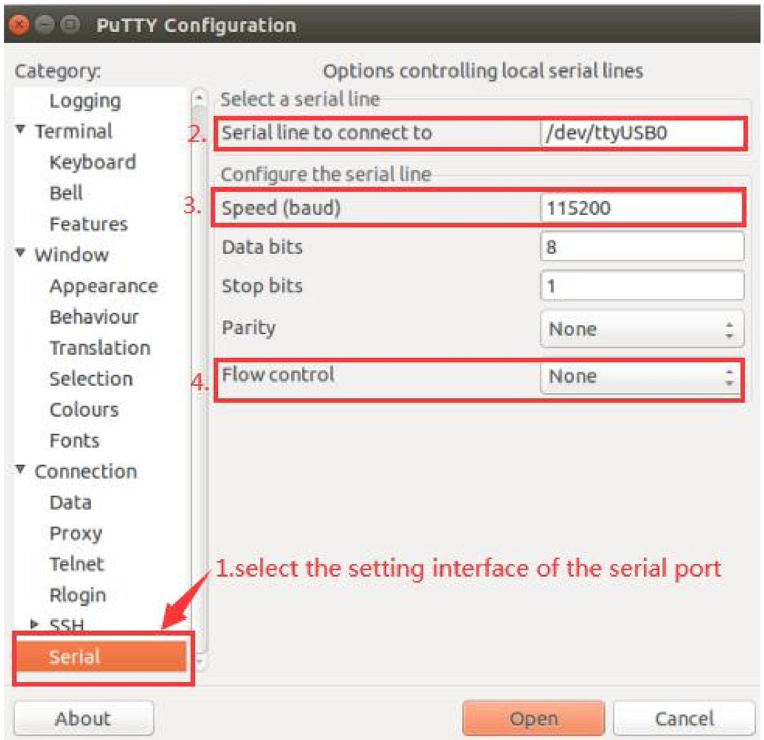


7) Then set the parameters of the serial port

a. Set Serial line to connect to to /dev/ttyUSB0 (modify to the corresponding node name, generally /dev/ttyUSB0)

b. Set Speed(baud) to 115200

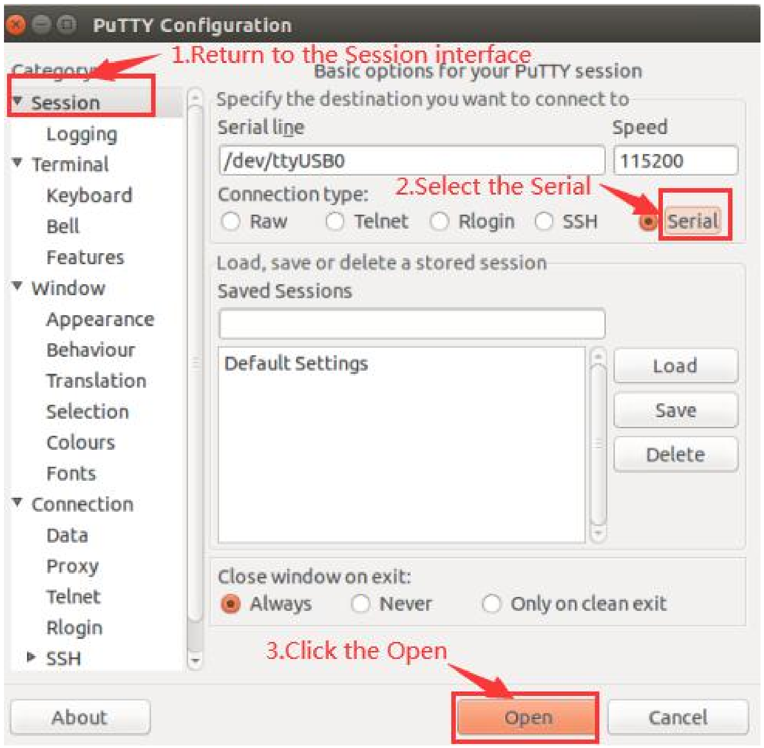
c. Set Flow control to None



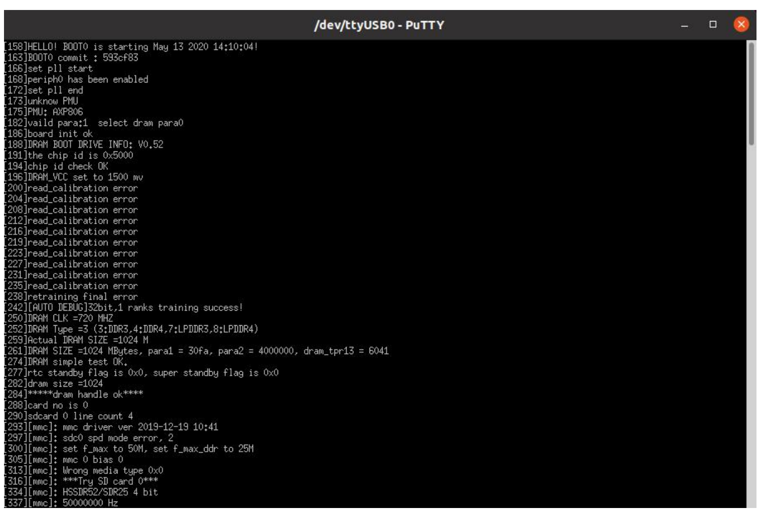
8) After setting the serial port setting interface, return to the Session interface

a. First select the Connection type as Serial

b. Then click the Open button to connect to the serial port



9) After starting the development board, you can see the Log information output by the system from the opened serial terminal



2. 7. 3. How to use the debug serial port on Windows platform

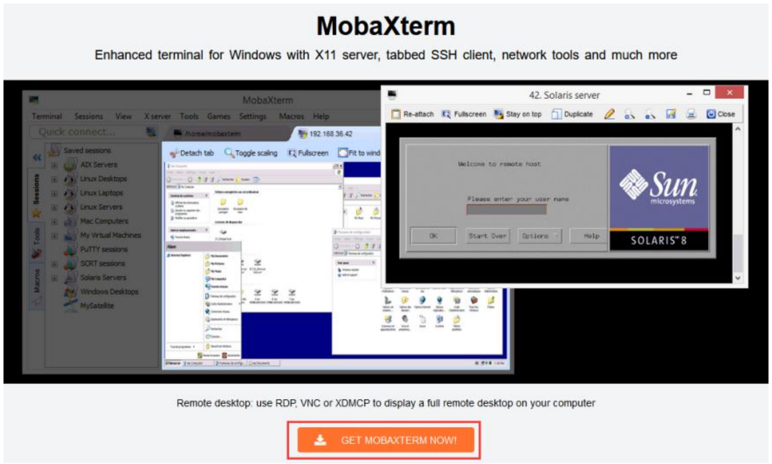
1) There are many serial debugging tools that can be used under Windows, such as SecureCRT, MobaXterm, etc. The following demonstrates how to use MobaXterm. This software is free and can be used without purchasing a serial number.

2) Download MobaXterm

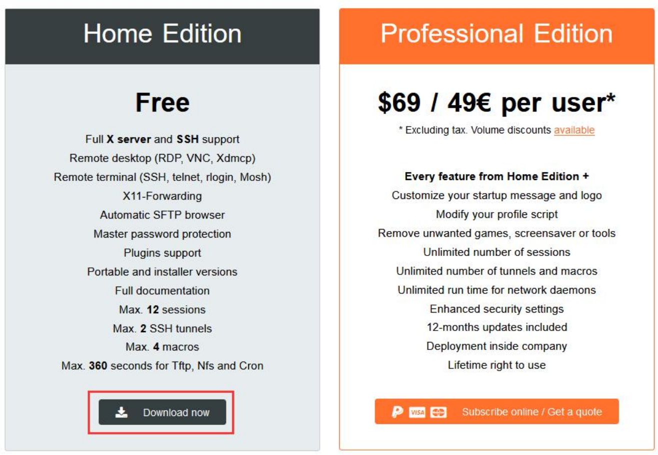
a. Download MobaXterm URL as follows

https://mobaxterm.mobatek.net/

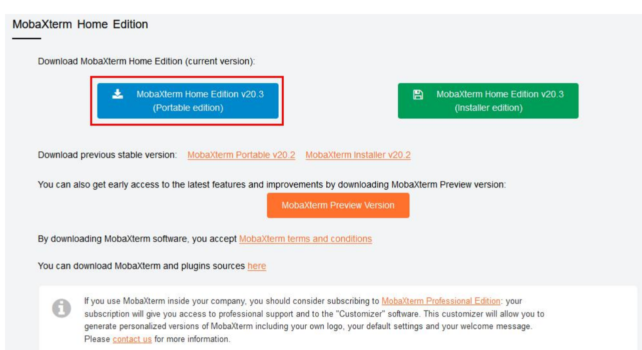
b. **After entering the MobaXterm download page, click GET XOBATERM NOW!**



c. Then choose to download the Home version



d. Then select the Portable version, after downloading, you don't need to install it, just open it and you can use it



3) After downloading, use the decompression software to decompress the downloaded compressed package, you can get the executable software of MobaXterm, and then double-click to open it

4) After opening the software, the steps to set the serial port connection are as follows

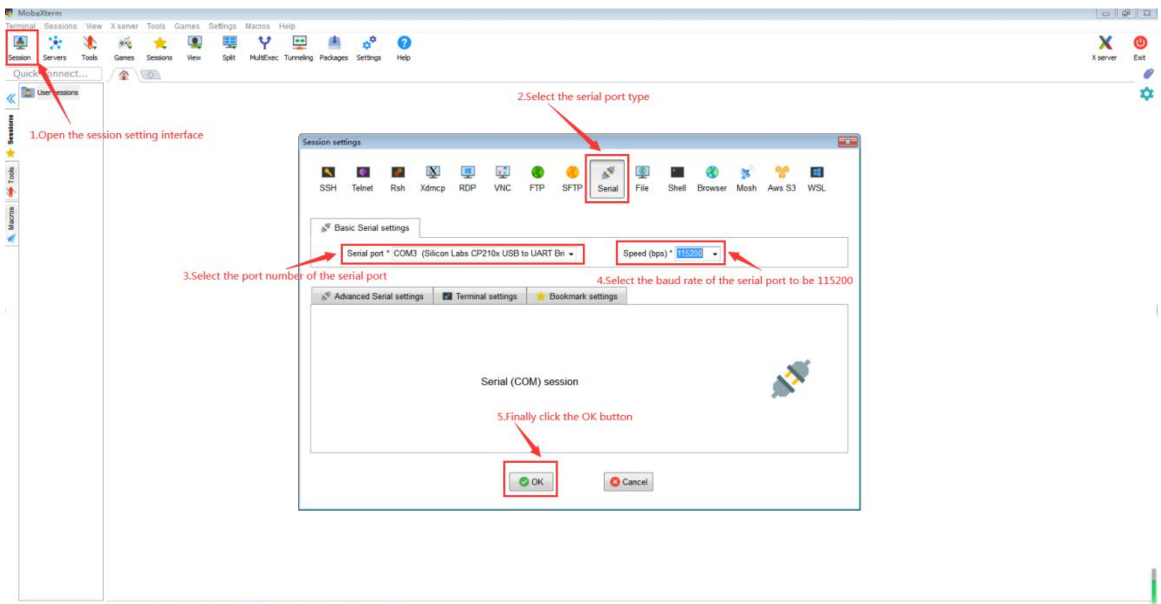
a. Open the session setting interface

b. Select the serial port type

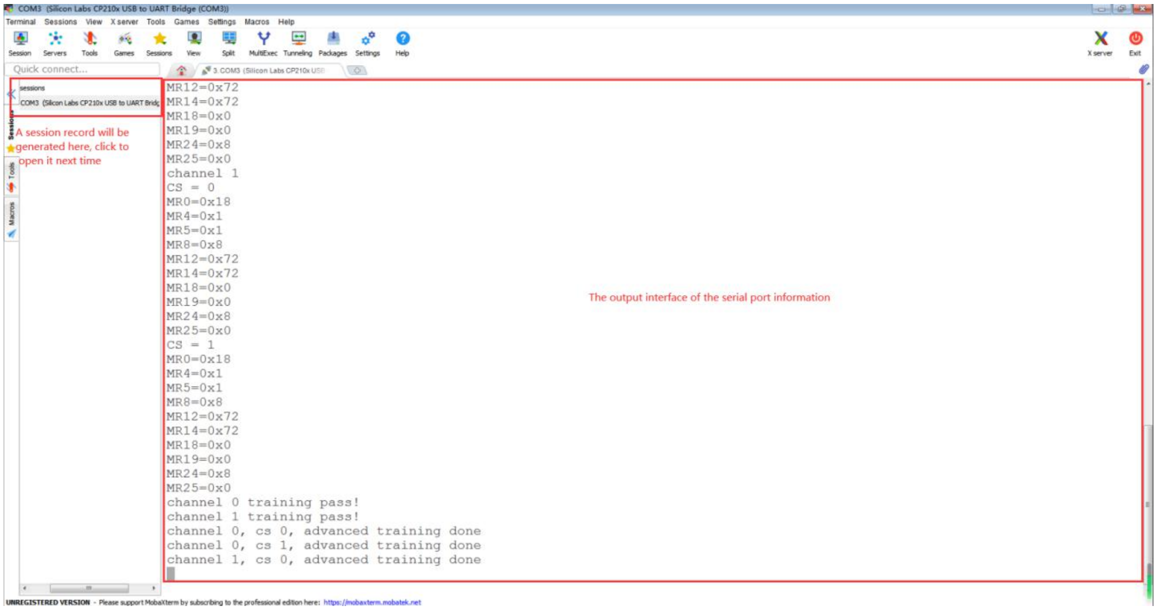
c. Select the port number of the serial port (choose the corresponding port number according to the specific situation), if you can't see the port number, please use the 360 driver master to scan and install the USB to TTL serial chip driver

d. Select the baud rate of the serial port to be 115200

e. Finally click the OK button to complete the setting



5) After clicking OK, you will enter the following interface, and you can see the output information of the serial port when you start the development board.



3. Linux system instructions

3. 1. Supported Linux distribution types and kernel versions

Release version	Kernel version	Server version	desktop version
Ubuntu 20.04	linux5.4	Support	Support
Ubuntu 18.04	linux5.4	Support	Support
Debian 10	linux5.4	Support	Support
Ubuntu 16.04	linux3.4	Support	Support

3. 2. Linux5.4 kernel image driver adaptation situation

Function	Status
HDMI video	OK
HDMI Audio	OK
USB2.0 x 3	OK
TF card boot	OK
Network card	OK
IR receiver	OK

Headphone Audio	OK
MIC recording	OK
USB camera	OK
LED	OK
40pin GPIO	OK
I2C	OK
SPI	OK
UART	OK
Temperature Sensor	OK
Hardware watchdog	OK
OV5640 camera	OK
GC2035 camera	NO
TV-OUT	NO

3. 3. Linux3.4 kernel image driver adaptation situation

Function	Status
HDMI video	OK
HDMI Audio	OK
USB2.0 x 3	OK
TF card boot	OK
Network card	OK
IR receiver	OK
Headphone Audio	OK
USB camera	OK
LED	OK
40pin GPIO	OK
I2C	OK
SPI	OK
UART	OK
Temperature Sensor	OK
Hardware watchdog	OK
OV5640 camera	OK

GC2035 camera	OK
Mali GPU	OK
TV-OUT	OK

3. 4. Login account and password

Account	Password
root	orangepi
orangepi	orangepi

3. 5. On-board LED light display control instructions

1) There are two LED lights on the development board, one green light and one red light. The default display of the LED lights when the system starts is as follows

	Green Light	Red Light
u-boot startup phase	Light off	Bright
Kernel boot to enter the system	Bright	Off or flash
GPIO port	PL10	PA15

2) The method of setting the green light on and off and flashing is as follows (take linux3.4 system as an example)

a. First enter the green light setting directory

```
root@orangepi:~# cd /sys/class/leds/green_led
```

b. The command to set the green light off is as follows

```
root@orangepi:/sys/class/leds/green_led# echo 0 > brightness
```

c. The command to set the green light is as follows

```
root@orangepi:/sys/class/leds/green_led# echo 1 > brightness
```

d. The command to set the green light to flash is as follows

```
root@orangepi:/sys/class/leds/green_led# echo heartbeat > trigger
```

e. The command to set the green light to stop flashing is as follows

```
root@orangepi:/sys/class/leds/green_led# echo none > trigger
```

3) The method of setting the red light on and off and flashing is as follows (take the

linux3.4 system as an example)

a. First enter the red light setting directory

```
root@orangepi:~# cd /sys/class/leds/red_led
```

b. The command to set the red light off is as follows

```
root@orangepi:/sys/class/leds/red_led# echo 0 > brightness
```

c. The command to set the red light to be always on is as follows

```
root@orangepi:/sys/class/leds/red_led# echo 1 > brightness
```

d. The command to set the red light to flash is as follows

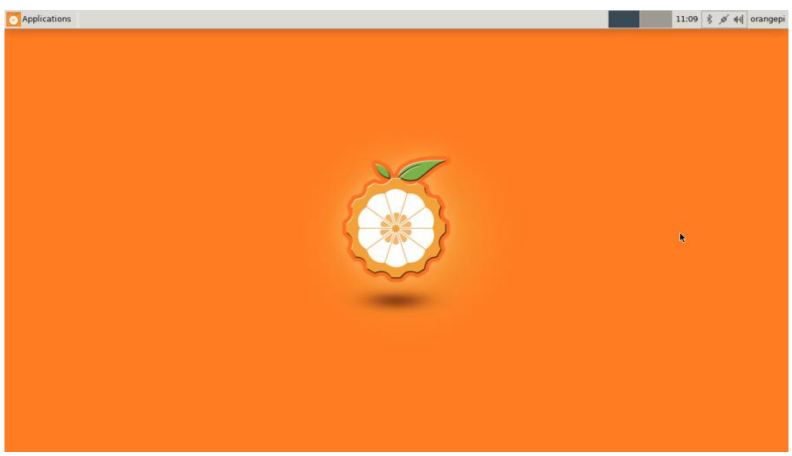
```
root@orangepi:/sys/class/leds/red_led# echo heartbeat > trigger
```

e. The command to set the red light to stop flashing is as follows

```
root@orangepi:/sys/class/leds/red_led# echo none > trigger
```

3. 6. Description of automatic login for desktop version system

1) The linux5.4 desktop version system will automatically log in to the desktop after it is started by default, without entering a password



2) Modify the configuration in **/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf** to prohibit the desktop version system from automatically logging in to the desktop. The modification command is as follows, or you can open the configuration file to modify it directly

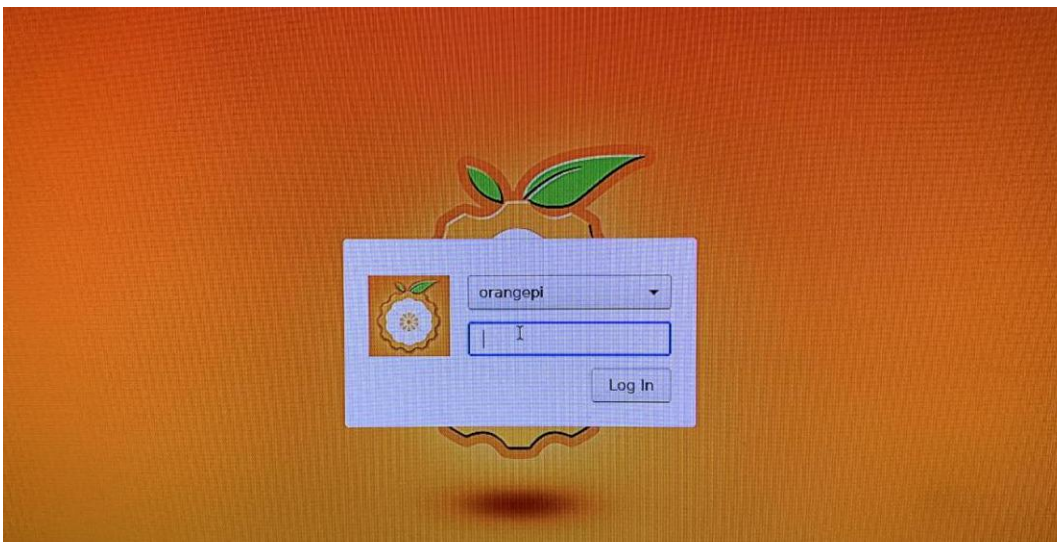
```
root@orangepis:~# sed -i "s/autologin-user=orangepi/#autologin-user=orangepi/" /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
```

3) After modification, the configuration of

/etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf is as follows

```
root@orangepis:~# cat /etc/lightdm/lightdm.conf.d/22-orangepi-autologin.conf
[Seat:*]
#autologin-user=orangepi
autologin-user-timeout=0
user-session=xfce
```

4) Then restart the system and a login dialog box will appear, at this time you need to enter a password to enter the system



3.7. The first time the Linux5.4 system starts to automatically expand rootfs

1) When the linux5.4 system is started for the first time through the TF card, the orangepi-resize-filesystem script will be called through the orangepi-resize-filesystem.service systemd service to automatically expand the rootfs

2) After logging in to the system, you can use the df -h command to check the size of rootfs. If it is consistent with the actual capacity of the TF card, it means that the automatic expansion is running correctly

```
root@orangepi:~# df -h
Filesystem            Size    Used Avail Use% Mounted on
udev                   430M        0    430M      0% /dev
```

```
tmpfs            100M  5.6M   95M   6% /run
/dev/mmcblk0p1    15G  915M   14G   7% /
tmpfs            500M     0  500M   0% /dev/shm
```

3) It should be noted that the Linux system has only one partition in ext4 format, and does not use a separate BOOT partition to store files such as kernel images, so there is no problem of BOOT partition expansion

4) In addition, if you do not need to automatically expand rootfs, you can use the following method to prohibit

a. First burn the linux image to the TF card
b. Then insert the TF card into the Ubuntu PC (Windows does not work), the Ubuntu PC will usually automatically mount the TF card partition. If the automatic mounting is normal, use the ls command to see the following output, the TF card partition name and the following command The names shown are not necessarily the same, please modify according to the actual situation

```
test@test:~$ ls /media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/
bin   boot   dev   etc   home   lib   lost+found   media   mnt   opt   proc   root   run
sbin   selinux   srv   sys   tmp   usr   var
```

c. Then switch the current user to root user in Ubuntu PC

```
test@test:~$ sudo -i
[sudo] test 的密码：
root@test:~#
```

d. Then enter the root directory of the Linux system in the TF card and create a new file named .no_rootfs_resize

```
root@test:~# cd /media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db# cd root
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/root# touch .no_rootfs_resize
root@test:/media/test/27e62f92-8250-4ef1-83db-3d8f0c2e23db/root# ls .no_rootfs*
.no_rootfs_resize
```

e. Then you can delete the TF card, then unplug the TF and plug it into the development board to start. When the linux system starts, when it detects that there is a file .no_rootfs_resize in the /root directory, the rootfs will no longer be automatically expanded
f. After disabling automatic expansion of rootfs, you can see that the available

capacity of the TF card is only about 200M

```
root@orangepi:~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            927M     0  927M   0% /dev
tmpfs           200M  5.6M  194M   3% /run
/dev/mmcblk0p1  1.5G  1.3G  196M  87% /
tmpfs           997M     0  997M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           997M     0  997M   0% /sys/fs/cgroup
tmpfs           997M  4.0K  997M   1% /tmp
/dev/zram0       49M  1.5M   44M   4% /var/log
tmpfs           200M     0  200M   0% /run/user/0
```

3. 8. Linux3.4 system automatic expansion rootfs instructions

1) When the linux3.4 system is started for the first time through the TF card, the orangepi-resize-filesystem script will be called through the orangepi-resize-filesystem.service systemd service to automatically expand the rootfs, but it is different from the linux5.4 system. After the first boot is completed, the automatic expansion has not been completed, and the system needs to be restarted to finally complete the automatic expansion of rootfs

2) When you start the linux3.4 system for the first time, you will see a warning when you log in to the system through ssh or serial port: a restart is required to complete the expansion of the file system, please restart as soon as possible

a. If you see this warning, please restart as soon as possible, and perform other operations after the automatic expansion is completed

```
  ___  ____  _
 / _ \|  _ \(_)
| | | | |_) | |
| |_| |  __/| |
 \___/|_|   |_|

Welcome to Orange Pi Xenial with Linux 3.4.113-sun8i

System load:   1.04 0.46 0.17   Up time:       1 min           Local users:   2

Memory usage:  16 % of [illegible]   IP:            192.168.1.143
CPU temp:      44°C
Usage of /:    84% of 2.0G

Warning: a reboot is needed to finish resizing the filesystem
Please reboot the system as soon as possible

New to Orange Pi? Support: http://www.orangepi.org

root@orangepi:~#
```

b. After starting the linux system for the first time, you can see the size of rootfs as shown below before restarting, only a few hundred megabytes of free space

```
root@orangepi:~# df -h
Filesystem        Size   Used Avail Use% Mounted on
udev               370M      0  370M    0% /dev
tmpfs              101M   2.1M   99M    3% /run
/dev/mmcblk0p1     2.0G   1.6G  335M   84% /
tmpfs              501M   140K  501M    1% /dev/shm
```

3) After restarting, log in to the system through ssh or serial port to see

a. The warning that needs to restart to complete the expansion has disappeared

```
  ___  ____  _
 / _ \|  _ \(_)
| | | | |_) | |
| |_| |  __/| |
 \___/|_|   |_|

Welcome to Orange Pi Xenial with Linux 3.4.113-sun8i

System load:   0.18 0.33 0.15   Up time:       3 min           Local users:   2

Memory usage:  15 % of 1000MB   IP:            192.168.1.143
CPU temp:      45°C
Usage of /:    12% of 15G

[ General system configuration (beta): orangepi-config ]

Last login: Thu Mar 11 09:22:16 2021 from 192.168.1.54

root@orangepi:~#
```

b. Use the df -h command to check the size of the rootfs. If the automatic expansion is running correctly, you can see that the size of the rootfs is basically the same as the actual capacity of the TF card

```
root@orangepi:~# df -h
Filesystem          Size   Used Avail Use% Mounted on
udev                430M      0  430M   0% /dev
tmpfs               100M   5.6M   95M   6% /run
/dev/mmcblk0p1      15G   915M   14G    7% /
tmpfs               500M      0  500M   0% /dev/shm
```

4) It should be noted that the linux3.4 system has only one partition in ext4 format, and does not use a separate BOOT partition to store files such as kernel images, so there is no problem of BOOT partition expansion

5) In addition, if you do not need to automatically expand rootfs, you can use the following method to prohibit

a. First burn the linux image to the TF card
b. Then insert the TF card into the Ubuntu PC (Windows does not work), the Ubuntu PC will usually automatically mount the TF card partition. If the automatic mounting is normal, use the ls command to see the following output, the TF card partition name and the following command The names shown are not necessarily the same, please modify according to the actual situation

```
test@test:~$ ls /media/test/49cc0cc0-8cb2-435d-bd35-4bbc6b7cd975/
bin    dev   home   lost+found   mnt   proc   run    selinux   sys   usr
boot   etc   lib    media        opt   root   sbin   srv       tmp   var
```

c. Then switch the current user to root user in Ubuntu PC

```
test@test:~$ sudo -i
[sudo] test 的密码：
root@test:~#
```

d. Then enter the root directory of the Linux system in the TF card and create a new file named .no_rootfs_resize

```
root@test:~# cd /media/test/49cc0cc0-8cb2-435d-bd35-4bbc6b7cd975
root@test:/media/test/49cc0cc0-8cb2-435d-bd35-4bbc6b7cd975# cd root
root@test:/media/test/49cc0cc0-8cb2-435d-bd35-4bbc6b7cd975/root# touch .no_rootfs_resize
root@test:/media/test/49cc0cc0-8cb2-435d-bd35-4bbc6b7cd975/root# ls .no_rootfs*
.no_rootfs_resize
```

e. Then you can unmount the TF card, then unplug the TF and plug it into the development board to start. When the linux system starts, when it detects that there is a file .no_rootfs_resize in the /root directory, the rootfs will no longer be automatically expanded

f. After disabling rootfs automatic expansion, after the first startup, you will no longer see the warning that you need to restart to complete expansion after logging in to the system through ssh or serial port. Even after restarting, you can see that the available capacity of the TF card is only about 300M

```
root@orangepi:~# df -h
Filesystem       Size  Used Avail Use% Mounted on
udev             370M     0  370M   0% /dev
tmpfs            101M  2.0M   99M   2% /run
/dev/mmcblk0p1   2.0G  1.6G  335M  84% /
tmpfs            501M  140K  501M   1% /dev/shm
```

3. 9. How to modify the linux log level (loglevel)

1) The loglevel of the linux system is set to 1 by default. When using the serial port to view the startup information, the kernel output log is as follows, basically all shielded

```
Starting kernel ...

Uncompressing Linux... done, booting the kernel.

Orange Pi 2.1.0 Bionic ttyS0

orangepi login:
```

2) When there is a problem with the linux system startup, you can use the following method to modify the value of loglevel, so as to print more log information to the serial port to display, which is convenient for debugging. If the Linux system fails to start and cannot enter the system, you can insert the TF card into the Ubuntu PC through a card reader, and then directly modify the configuration of the linux system in the TF card after mounting the TF card in the Ubuntu PC. Insert the TF card into the development board to start

```
root@orangepi:~# sed -i "s/verbosity=1/verbosity=7/" /boot/orangepiEnv.txt
```

```
root@orangepi:~# sed -i "s/console=both/console=serial/" /boot/orangepiEnv.txt
```

3) The above commands are actually to set the variables in /boot/orangepiEnv.txt. After setting, you can open /boot/orangepiEnv.txt to check

```
root@orangepi:~# cat /boot/orangepiEnv.txt
verbosity=7
bootlogo=false
console=serial
```

4) Then restart the development board, the output information of the kernel will be printed to the serial port for output

```
[  OK  ] Started Dispatcher daemon for systemd-networkd.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Started Update UTMP about System Runlevel Changes.
[  OK  ] Started Authorization Manager.

Orange Pi 2.1.0 Bionic ttyS0

orangepi login:
```

3. 10. SSH remote login to the development board

Linux systems have SSH remote login enabled by default, and allow root users to log in to the system. Before ssh login, you need to make sure that the Ethernet or wifi network is connected, and then use the ifconfig command or check the router to obtain the IP address of the development board

3. 10. 1. SSH remote login development board under Ubuntu

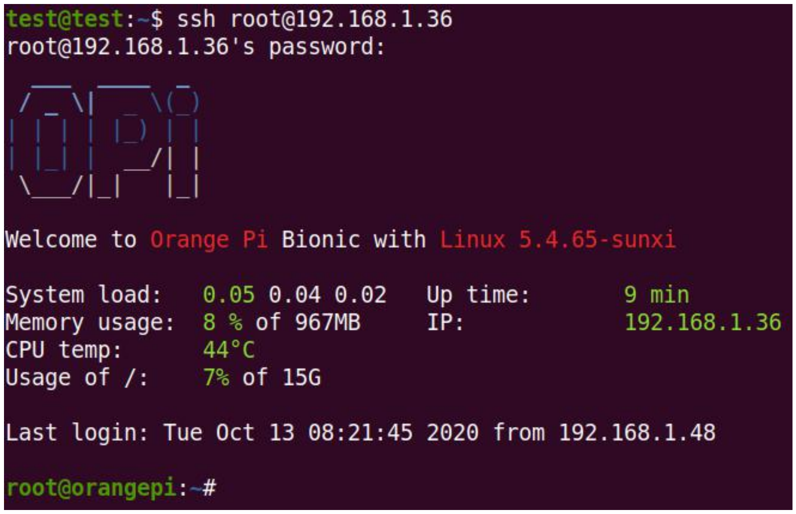
1) First get the IP address of the development board

2) Then you can log in to the linux system remotely through the ssh command

```
test@test:~$ ssh root@192.168.1.36        // Need to be replaced with the IP address of the development board
root@192.168.1.36's password:               // Enter the password here, the default
```

password is orangepi

3) The display after successfully logging in to the system is as shown in the figure below



4) If the following error is prompted when ssh login

test@test:~$ **ssh root@192.168.1.36**
Connection reset by 192.168.1.149 port 22
lost connection

You can enter the following command on the development board and try to connect

root@orangepi:~# **rm /etc/ssh/ssh_host_***
root@orangepi:~# **dpkg-reconfigure openssh-server**

3.10.2. SSH remote login development board under Windows

1) First get the IP address of the development board

2) In windows, you can use MobaXterm to remotely log in to the development board, first create a new ssh session

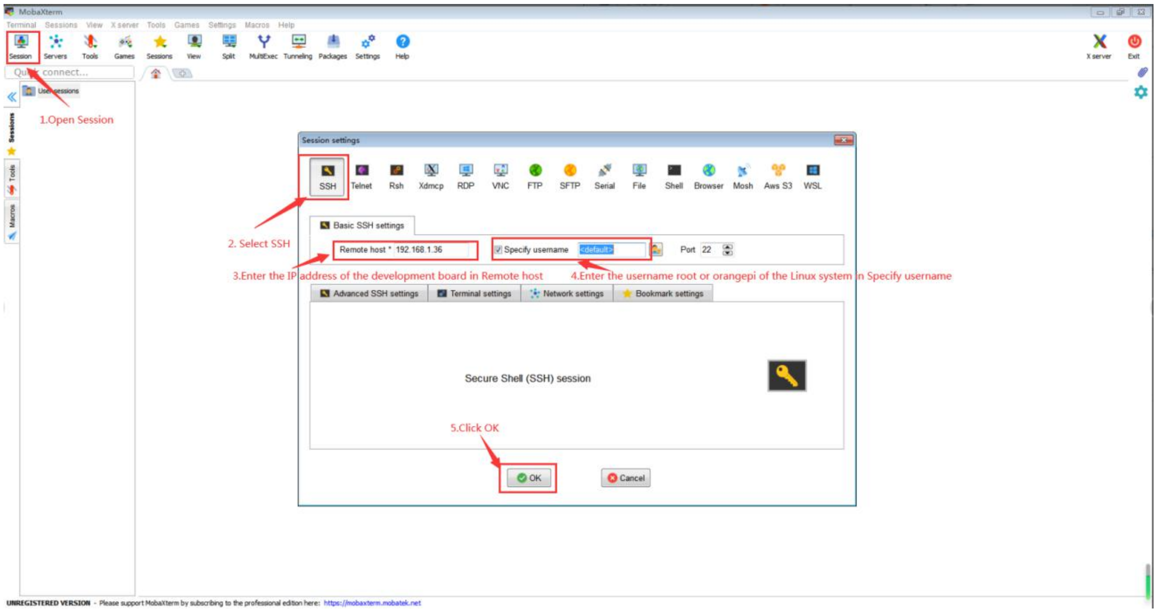
a. Open **Session**

b. Then select **SSH** in **Session Setting**

c. Then enter the IP address of the development board in **Remote host**

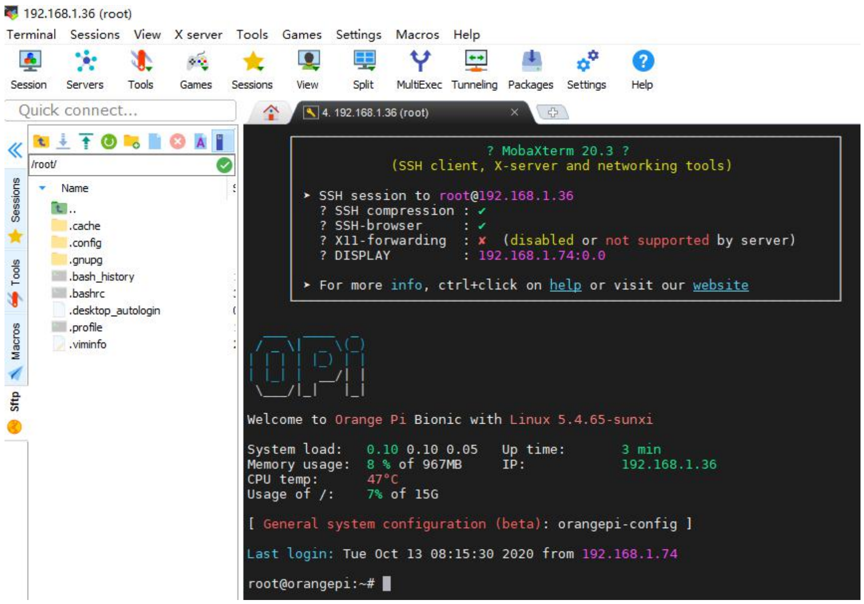
d. Then enter the username **root** or **orangepi** of the Linux system in **Specify username**

e. Finally click **OK**



3) Then you will be prompted to enter a password, the default passwords for both root and orangepi users are orangepi

4) The display after successfully logging in to the system is as shown in the figure below



3. 11. Ethernet port test

1) First, insert the network cable into the Ethernet interface of the development board, and ensure that the network is unblocked

2) After the system starts, it will automatically assign an IP address to the Ethernet card through DHCP

3) The command to view the IP address is as follows

```
root@orangepi:~# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.47  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::e56:c34d:62f0:8d6e  prefixlen 64  scopeid 0x20<link>
        ether 02:81:3e:a8:58:d8  txqueuelen 1000  (Ethernet)
        RX packets 2165  bytes 177198 (177.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 312  bytes 40435 (40.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 39
```

4) The command to test network connectivity is as follows

```
root@orangepi:~# ping www.orangepi.org -I eth0
PING www.orangepi.org (182.92.236.130) from 192.168.1.47 eth0: 56(84) bytes of data.
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=1 ttl=52 time=39.3 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=2 ttl=52 time=39.3 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=3 ttl=52 time=39.9 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=4 ttl=52 time=39.7 ms
64 bytes from 182.92.236.130 (182.92.236.130): icmp_seq=5 ttl=52 time=39.7 ms
^C
--- www.orangepi.org ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 39.323/39.617/39.922/0.274 ms
```

3. 12. HDMI display test

1) Use HDMI to HDMI cable to connect Orange Pi development board and HDMI display

2) If the HDMI display has image output after starting the linux system, it means that the HDMI interface is in normal use

3) In the absence of network and serial port, you can use HDMI display, and then connect the mouse and keyboard to control the development board

3. 13. USB interface test

3. 13. 1. Connect mouse or keyboard test

1) Insert the keyboard of the USB interface into the USB interface of the Orange Pi

development board

2) Connect the Orange Pi development board to the HDMI display

3) If the mouse or keyboard can operate normally, the USB interface is used normally (the mouse can only be used in the desktop version image)

3. 13. 2. Connect USB storage device test

1) Format the U disk first, and then put some files in the U disk

2) Then insert the U disk into the USB interface of the development board

3) Execute the following command, if you can see the output of sdX, it means that the U disk has been recognized successfully

```
root@orangepi:~# cat /proc/partitions | grep "sd*"
major minor  #blocks  name
   8        0   30044160 sda
   8        1   30043119 sda1
```

4) Use the mount command to mount the U disk to /mnt, and then you can view the files in the U disk

```
root@orangepi:~# mount /dev/sda1 /mnt/
root@orangepi:~# ls /mnt/
test.txt
```

5) After mounting, you can view the capacity usage and mount point of the U disk through the df command

```
root@orangepi:~# df -h | grep "sd"
/dev/sda1         29G   208K    29G    1% /mnt
```

3. 14. USB Ethernet card test

1) The USB Ethernet cards that have been tested and can be used are as follows. Among them, the RTL8153 USB Gigabit network card is inserted into the USB 2.0 Host interface of the development board. The test can be used normally, but the speed is not up to

Gigabit. Please pay attention to this point.

Serial number	model
1	RTL8152B USB 100M network card
2	RTL8153 USB Gigabit Ethernet

2) First insert the USB network card into the USB interface of the development board, and then insert the network cable into the USB network card to ensure that the network cable can normally access the Internet. If you can see the following log information through the dmesg command, it means that the USB network card is recognized normally

```
root@orangepi:~# dmesg | tail
[   121.985016] usb 3-1: USB disconnect, device number 2
[   126.873772] sunxi-ehci 5311000.ehci3-controller: ehci_irq: highspeed device connect
[   127.094054] usb 3-1: new high-speed USB device number 3 using sunxi-ehci
[   127.357472] usb 3-1: reset high-speed USB device number 3 using sunxi-ehci
[   127.557960] r8152 3-1:1.0 eth1: v1.08.9
[   127.602642] r8152 3-1:1.0 enx00e04c362017: renamed from eth1
[   127.731874] IPv6: ADDRCONF(NETDEV_UP): enx00e04c362017: link is not ready
[   127.763031] IPv6: ADDRCONF(NETDEV_UP): enx00e04c362017: link is not ready
[   129.892465] r8152 3-1:1.0 enx00e04c362017: carrier on
[     129.892583]  IPv6:  ADDRCONF(NETDEV_CHANGE):  enx00e04c362017:  link
becomes ready
```

3) Then you can see the device node of the USB network card and the automatically assigned IP address through the ifconfig command

```
root@orangepi:~# ifconfig
enx00e04c362017:   flags=4163<UP,BROADCAST,RUNNING,MULTICAST>       mtu
1500
          inet 192.168.1.177    netmask 255.255.255.0    broadcast 192.168.1.255
          inet6 fe80::681f:d293:4bc5:e9fd    prefixlen 64    scopeid 0x20<link>
          ether 00:e0:4c:36:20:17    txqueuelen 1000    (Ethernet)
          RX packets 1849    bytes 134590 (134.5 KB)
          RX errors 0    dropped 125    overruns 0    frame 0
          TX packets 33    bytes 2834 (2.8 KB)
          TX errors 0    dropped 0 overruns 0    carrier 0    collisions 0
```

4) The command to test network connectivity is as follows

```
root@orangepi:~# ping www.baidu.com -I enx00e04c362017
PING www.a.shifen.com (14.215.177.38) from 192.168.1.12 eth0: 56(84) bytes of data.
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=1 ttl=56 time=6.74 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=2 ttl=56 time=6.80 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=3 ttl=56 time=6.26 ms
64 bytes from 14.215.177.38 (14.215.177.38): icmp_seq=4 ttl=56 time=7.27 ms
^C
--- www.a.shifen.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 6.260/6.770/7.275/0.373 ms
```

3.15. USB camera test

1) First insert the USB camera into the USB port of the Orange Pi development board

2) Use lsmod to check whether the system has automatically loaded the uvcvideo kernel module

```
root@orangepi:~# lsmod
Module                  Size  Used by
uvcvideo              106496  0
```

3) Then use v4l2-ctl (note that l in v4l2 is a lowercase letter l, not a number 1) command to view the device node of the USB camera. From the output below, you can see that the device node corresponding to the USB camera is **/dev/video2**. If you look The USB-related video node is not found, indicating that the USB camera cannot be recognized

```
root@orangepi:~# apt update
root@orangepi:~# apt install v4l-utils
root@orangepi:~# v4l2-ctl   --list-devices
sun6i-csi (platform:camera):
        /dev/video1

cedrus (platform:cedrus):
        /dev/video0
```

```
USB 2.0 Camera: HD USB Camera (usb-1c1c000.usb-1):
        /dev/video2
        /dev/video3
```

4) Install fswebcam

```
root@orangepi:~# apt update
root@orangepi:~# apt-get install fswebcam
```

5) After installing fswebcam, you can use the following command to take pictures
   a. The -d option is used to specify the device node of the USB camera
   b. --no-banner is used to remove watermark from photos
   c. The -r option is used to specify the resolution of the photo
   d. -S option is used to skip the previous frame number

```
root@orangepi:~# fswebcam -d /dev/video2 --no-banner -r 1280x720 -S 5 ./image.jpg
```

6) In the server version of the Linux system, you can use the scp command to transfer the taken pictures to the image of Ubuntu PC after taking pictures.

```
root@orangepi:~# scp image.jpg test@192.168.1.55:/home/test // Need to be modified to the corresponding path
```

7) In the desktop version of the linux system, you can directly view the captured pictures through the HDMI display

3. 16. Audio test

3. 16. 1. Headphone jack play audio test

1) Through the aplay -l command, you can view the sound card devices supported by the system, where card 0 is the sound card device of the headset

```
root@orangepi:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: Codec [H3 Audio Codec], device 0: CDC PCM Codec-0 [CDC PCM Codec-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: allwinnerhdmi [allwinner-hdmi], device 0: 1c22800.i2s-i2s-hifi i2s-hifi-0 [1c22800.i2s-i2s-hifi i2s-hifi-0]
```

```
Subdevices: 1/1
Subdevice #0: subdevice #0
```

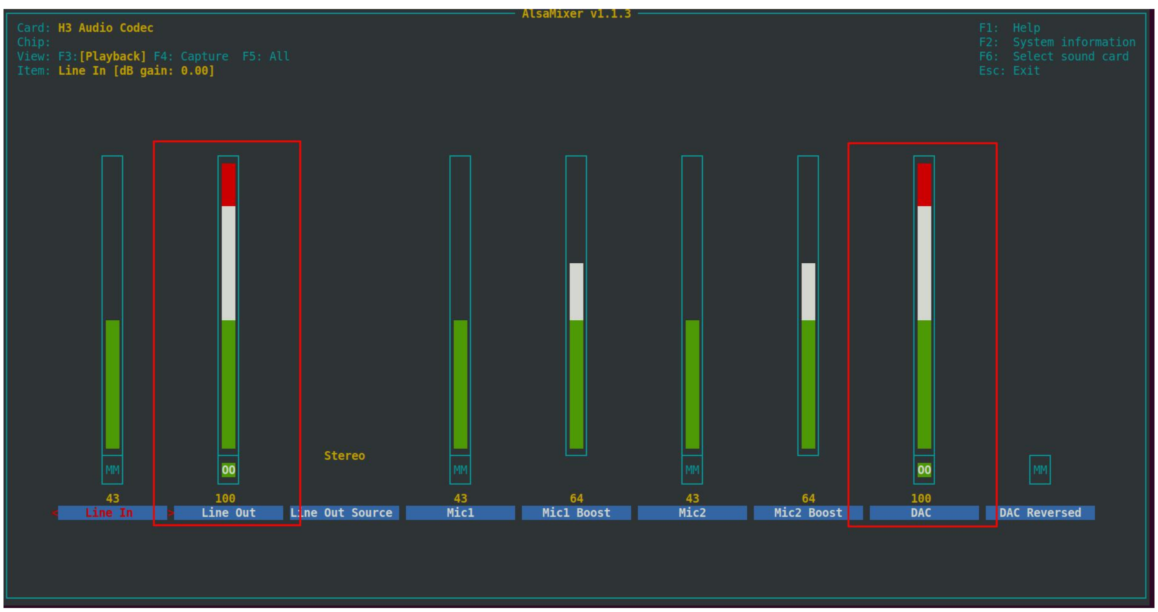
2) Upload the audio file that needs to be played to the /root folder of the Linux system. You can use the scp command to upload in the Ubuntu PC. The IP address in the command is the IP address of the Orange Pi development board and needs to be replaced

```
test@test:~/AudioTest$ scp audio.wav    root@192.168.1.47:/root
```

3) Then use alsamixer to open the audio adjustment interface (press **Esc** to exit alsamixer)

```
root@orangepi:~# alsamixer
```

a. The linux5.4 system needs to use the arrow keys to adjust the Line Out and DAC to the maximum



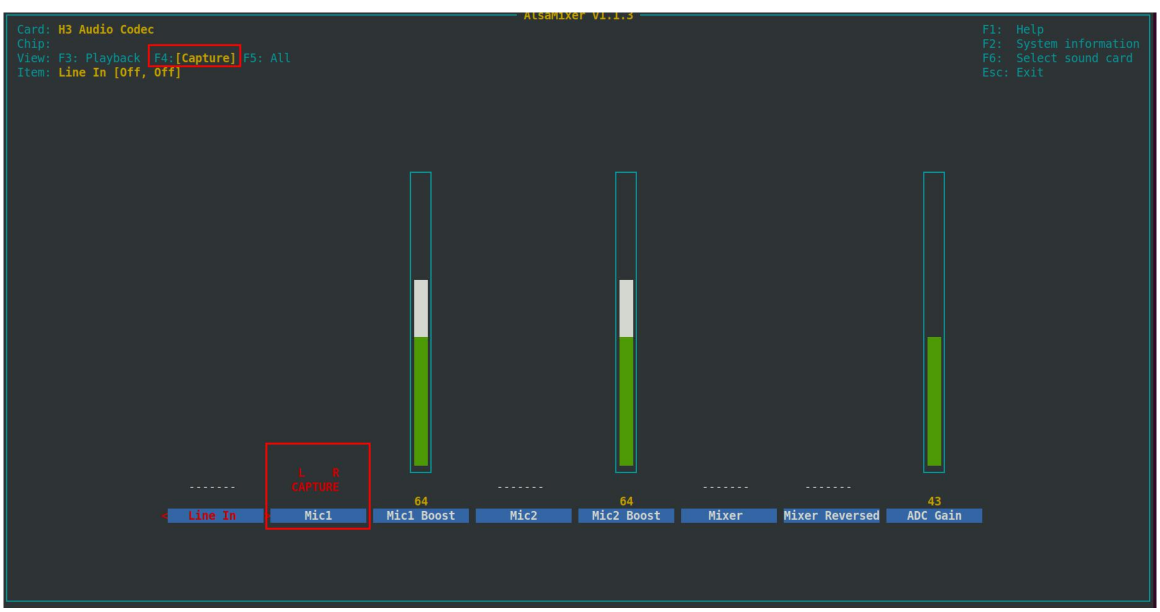
b. The linux3.4 system first needs to use the arrow keys to move the cursor to the **Audio lineout**, and then press the "**M**" key to open the **Audio lineout**. After opening, the box on the **Audio lineout** will display **00**

4) Then use the aplay command to play the audio headset to hear the sound

```
root@orangepi:~# aplay -D hw:0,0 audio.wav
```

3. 16. 2. Mic recording test

1) The linux5.4 system needs to use alsamixer to open the audio adjustment interface, then press the **F4** key to enter the recording channel setting, then use the arrow keys to move to the **Mic1** option, and then press the space bar to open the recording channel. **The linux3.4 system does not need to be set**



2) After opening the recording channel, you can start recording. The recording commands are as follows

```
root@orangepi:~# arecord -d 5 -f cd -t wav test.wav
Recording WAVE test.wav' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
root@orangepi:~#
```

3) After the recording is completed, a recording file named test.wav will be generated in the current path. Use the aplay command to play test.wav to check whether there is sound output

3. 16. 3. HDMI audio playback test

1) First use the aplay -l command to ensure that you can see the HDMI sound card device, where **card 1** is the HDMI sound card device

```
root@orangepi:~# aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: Codec [H3 Audio Codec], device 0: CDC PCM Codec-0 [CDC PCM Codec-0]
   Subdevices: 1/1
   Subdevice #0: subdevice #0
card 1: allwinnerhdmi [allwinner-hdmi], device 0: 1c22800.i2s-i2s-hifi i2s-hifi-0 [1c22800.i2s-i2s-hifi i2s-hifi-0]
   Subdevices: 1/1
   Subdevice #0: subdevice #0
```

2) HDMI audio playback does not require other settings, just use the aplay command to play directly

```
root@orangepi:~# aplay -D hw:1,0 audio.wav
```

3. 17. linux5.4 IR receiving test

1) Install ir-keytable IR test software

```
root@orangepi:~# apt update
root@orangepi:~# apt-get install ir-keytable
```

2) Then execute ir-keytable to view the information of the IR device

```
root@orangepi:~# ir-keytable
Found /sys/class/rc/rc0/ (/dev/input/event5) with:
     Name: sunxi-ir
```

```
Driver: sunxi-ir, table: rc-empty
lirc device: /dev/lirc0
Supported protocols: other lirc rc-5 rc-5-sz jvc sony nec sanyo mce_kbd rc-6 sharp xmp
Enabled protocols: lirc
bus: 25, vendor/product: 0001:0001, version: 0x0100
Repeat delay = 500 ms, repeat period = 125 ms
```

3) An IR remote control needs to be prepared before testing the IR receiving function

3. 18. linux3.4 IR receiving test

1) First load the **sunxi-cir** kernel module

```
root@orangepi:~# modprobe sunxi-cir
root@orangepi:~# lsmod
Module                  Size  Used by
ir_lirc_codec           3610  0
lirc_dev                7878  1 ir_lirc_codec
ir_mce_kbd_decoder      2957  0
ir_sanyo_decoder        1592  0
ir_sony_decoder         1446  0
ir_jvc_decoder          1532  0
ir_rc6_decoder          1997  0
ir_rc5_decoder          1444  0
ir_nec_decoder          1692  0
sunxi_cir               1609  0
```

```
rc_core                    12714   10
ir_lirc_codec,ir_rc5_decoder,ir_nec_decoder,ir_sony_decoder,sunxi_cir,ir_mce_kbd_dec
oder,ir_jvc_decoder,ir_rc6_decoder,ir_sanyo_decoder
```

2) Then install ir-keytable IR test software

```
root@orangepi:~# apt update
root@orangepi:~# apt-get install ir-keytable
```

3) Then execute ir-keytable to view the information of the IR device

```
root@orangepi:~# ir-keytable
Found /sys/class/rc/rc0/ (/dev/input/event6) with:
    Driver sunxi-ir, table rc-empty
    Supported protocols: lirc rc-5 jvc sony nec sanyo mce-kbd rc-6
    Enabled protocols: lirc rc-5 jvc sony nec sanyo mce-kbd rc-6
    Name: sunxi-ir
    bus: 25, vendor/product: 0001:0001, version: 0x0100
    Repeat delay = 500 ms, repeat period = 125 ms
```

4) An IR remote control needs to be prepared before testing the IR receiving function

4) Enter the **ir-keytable -t** command in the terminal, and then use the IR remote control to press the button against the IR receiving head of the Orange Pi development board to see the received key code in the terminal

```
root@orangepi:/# ir-keytable -t
Testing events. Please, press CTRL-C to abort.
1602840581.225093: event type EV_MSC(0x04): scancode = 0x45c
1602840581.225105: event type EV_SYN(0x00).
```

```
1602840581.663187: event type EV_MSC(0x04): scancode = 0x45c
1602840581.663197: event type EV_SYN(0x00).
```

3. 19. Hardware watchdog test

1) Download the code of wiringOP

```
root@orangepi:~# apt update
root@orangepi:~# apt install git
root@orangepi:~# git clone https://github.com/orangepi-xunlong/wiringOP
```

2) Compile wiringOP

```
root@orangepi:~# cd wiringOP
root@orangepi:~/wiringOP# ./build clean
root@orangepi:~/wiringOP# ./build
```

3) Compile the watchdog test program

```
root@orangepi:~/wiringOP# cd examples/
root@orangepi:~/wiringOP/examples# make watchdog
[CC] watchdog.c
[link]
```

4) Run the watchdog test program

a. The second parameter 10 represents the counting time of the watchdog. If the dog is not fed within this time, the system will restart

b. We can feed the dog by pressing any key on the keyboard (except ESC). After feeding the dog, the program will print a line of keep alive to indicate the success of feeding the dog

```
root@orangepi:~/wiringOP/examples# ./watchdog 10
open success
options is 33152,identity is sunxi-wdt
put_usr return,if 0,success:0
The old reset time is: 16
return ENOTTY,if -1,success:0
return ENOTTY,if -1,success:0
put_user return,if 0,success:0
```

```
put_usr return,if 0,success:0
keep alive
keep alive
```

3. 20. CSI camera test

3. 20. 1. CSI camera interface specifications

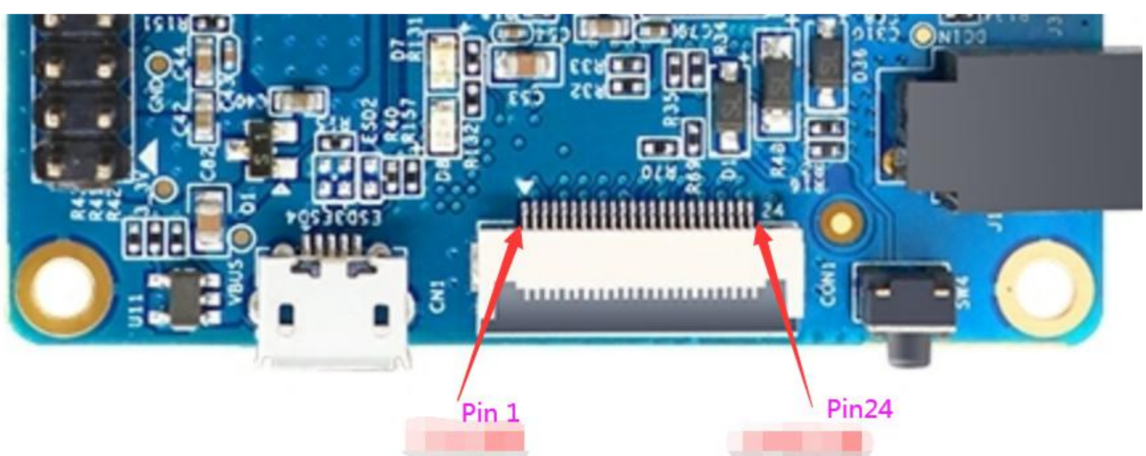
1) The CSI interface of Orange Pi PC supports two cameras, **gc2035** and **ov5640**. The support for cameras in different Linux systems is explained as follows

	GC2035	OV5640
linux3.4	Support	Support
linux5.4	Not Support	Support

2) The serial numbers of the CSI interface pins are shown in the figure below

a. The No. 1 pin of the CSI interface is connected to the No. 24 pin of the camera adapter board

b. The 24th pin of the CSI interface is connected to the 1st pin of the camera adapter board



3) CSI interface pins are defined as follows

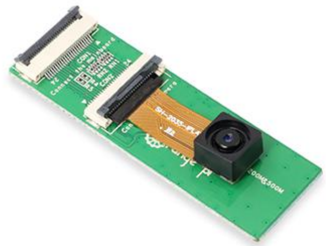
Pin	Function	GPIO port
CON1-P01	DCIN-5V	
CON1-P02	GND	
CON1-P03	TWI2-SDA	PE13
CON1-P04	CSI-PWR-EN	PA17
CON1-P05	TWI2-SCK	PE12

CON1-P06	CSI-RESET	PE15
CON1-P07	CSI-VSYNC	PE3
CON1-P08	CSI-STBY-EN	PE15
CON1-P09	CSI-HSYNC	PE2
CON1-P10	VDD1V8-CSI	PG11
CON1-P11	AFCC_EN	PG13
CON1-P12	CSI-D7	PE11
CON1-P13	CSI-MCLK	PE1
CON1-P14	CSI-D6	PE10
CON1-P15	GND	
CON1-P16	CSI-D5	PE9
CON1-P17	CSI-PCLK	PE0
CON1-P18	CSI-D4	PE8
CON1-P19	CSI-D0	PE4
CON1-P20	CSI-D3	PE7
CON1-P21	CSI-D1	PE5
CON1-P22	CSI-D2	PE6
CON1-P23	GND	
CON1-P24	DCIN-5V	

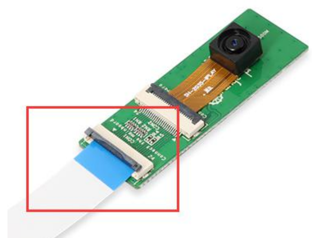
3. 20. 2. Linux3.4 system gc2035 camera test

1) The Gc2035 camera kit includes a gc2035 camera, an adapter board and a cable

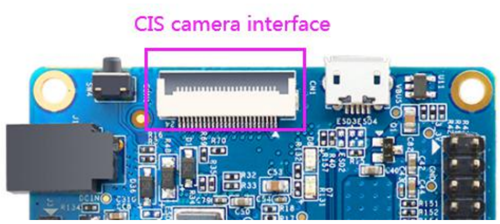
2) First insert the gc2035 camera into the adapter board

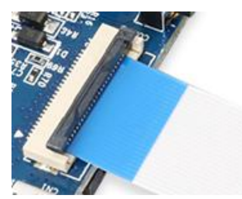


3) Then insert the ribbon cable into another card slot of the adapter board



4) Then insert the other end of the cable into the CIS camera interface of the development board. Start the linux system after connecting the camera (don't plug in the camera after power-on)





5) Then use the setsystem gc2035 command to initialize the configuration of gc2035

```
root@orangepi:~# setsystem gc2035
Set the csi camera used by the orangepipc as: gc2035
```

setsystem gc2035 The main things that the command does are:

a. Configure the kernel modules that need to be loaded for gc2035

```
root@orangepi:~# cat /etc/modules-load.d/modules.conf
gc2035
vfe_v4l2
```

b. Configure **vip_dev0_mname** in **/boot/script.bin** as gc2035

6) After restarting and entering the system, first confirm whether the kernel module

related to the gc2035 camera is automatically loaded

```
root@orangepi:~# lsmod
Module                  Size  Used by
vfe_v4l2             1018545  0
videobuf_dma_contig     3513  1 vfe_v4l2
videobuf_core          14871  2 vfe_v4l2,videobuf_dma_contig
gc2035                 19692  0
vfe_subdev              4531  2 vfe_v4l2,gc2035
cci                    22869  2 vfe_v4l2,gc2035
vfe_os                  4269  3 cci,vfe_v4l2,vfe_subdev
```

7) Then use v4l2-ctl (note that l in v4l2 is a lowercase letter l, not a number 1) command to view the device node of the CSI camera. From the output below, we can see that the device node corresponding to the camera is **/dev/video0**

```
root@orangepi:~# apt update
root@orangepi:~# apt install v4l-utils
root@orangepi:~# v4l2-ctl    --list-devices
sunxi-vfe (sunxi_vfe sunxi_vfe.0):
     /dev/video0
```

8) Then start to install the camera test software motion

```
root@orangepi:~# apt update
root@orangepi:~# apt install motion
```

9) Modify the configuration of /etc/default/motion, change start_motion_daemon=no to start_motion_daemon=yes

```
root@orangepi:~# sed -i "s/start_motion_daemon=no/start_motion_daemon=yes/" \
/etc/default/motion
```

10) Modify the configuration of /etc/motion/motion.conf

```
 root@orangepi:~# sed -i "s/stream_localhost on/stream_localhost off/" \
/etc/motion/motion.conf
```

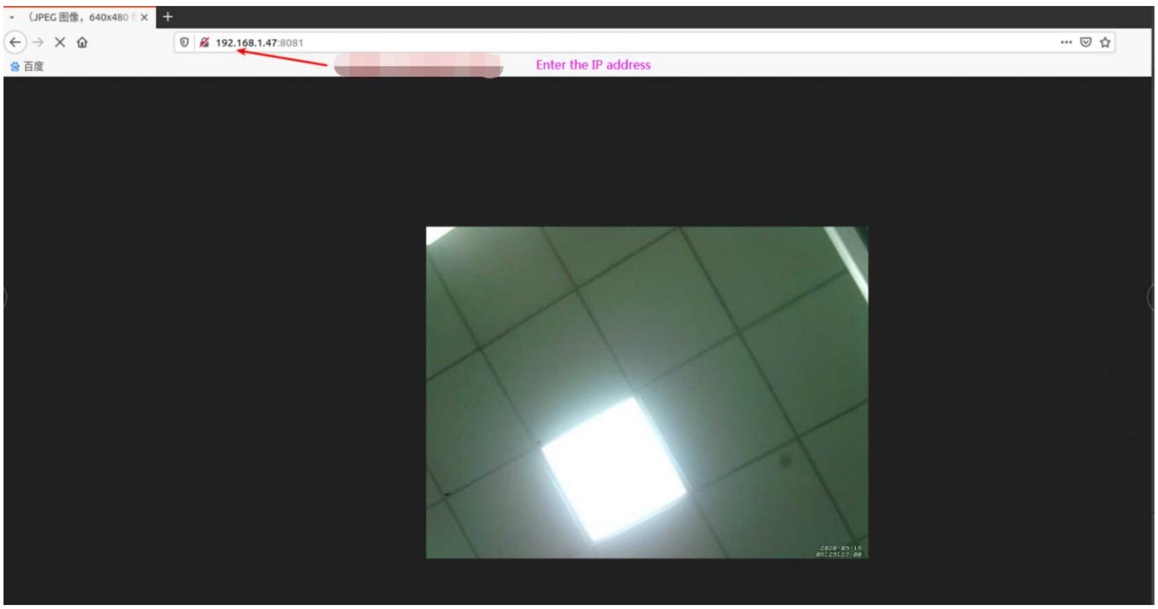
11) Then restart the motion service

```
root@orangepi:~# /etc/init.d/motion restart
```

```
[ ok ] Restarting motion (via systemctl): motion.service.
root@orangepi:~#
```

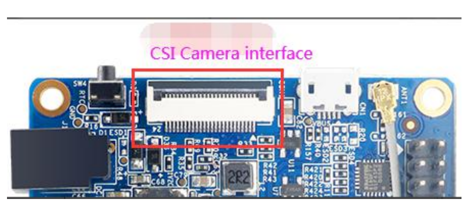
12) Before using motion, please make sure that the Orange Pi development board can be connected to the network normally, and then obtain the IP address of the development board through the ifconfig command

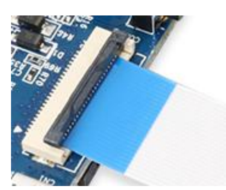
13) Then enter [development board IP address: 8081] in the Firefox browser to see the image output by gc2035



3. 20. 3. Linux3.4 system ov5640 camera test

1) First connect the Ov5640 camera adapter board to the CIS camera interface of the development board through a cable, and then start the linux system after connecting the camera (don’t plug in the camera after powering on)





2) Then use the **setsystem ov5640** command to initialize the configuration of ov5640

```
root@orangepi:~# setsystem ov5640
Set the csi camera used by the orangepipc as: ov5640
```

setsystem ov5640 The main things that the command does are：

a. Configure the kernel modules that need to be loaded for ov5640

```
root@orangepi:~# cat /etc/modules-load.d/modules.conf
ov5640
vfe_v4l2
```

b. Configure **vip_dev0_mname** in /**boot/script.bin** as ov5640

3) After restarting and entering the system, first confirm whether the kernel module related to the ov5640 camera is automatically loaded

```
root@orangepi:~# lsmod
Module                  Size  Used by
vfe_v4l2             1018545  1
videobuf_dma_contig     3513  1 vfe_v4l2
videobuf_core          14871  2 vfe_v4l2,videobuf_dma_contig
ov5640                 42317  0
vfe_subdev              4531  2 vfe_v4l2,ov5640
cci                    22869  2 vfe_v4l2,ov5640
vfe_os                  4269  3 cci,vfe_v4l2,vfe_subdev
```

4) Then use v4l2-ctl (note that l in v4l2 is a lowercase letter l, not a number 1) command to view the device node of the CSI camera. From the output below, we can see that the device node corresponding to the camera is **/dev/video0**

```
root@orangepi:~# apt update
root@orangepi:~# apt install -y  v4l-utils
root@orangepi:~# v4l2-ctl  --list-devices
sunxi-vfe (sunxi_vfe sunxi_vfe.0):
    /dev/video0
```

5) Then start to install the camera test software motion

```
root@orangepi:~# apt update
root@orangepi:~# apt install -y motion
```

6) Modify the configuration of **/etc/default/motion**, change **start_motion_daemon=no** to **start_motion_daemon=yes**

```
root@orangepi:~# sed -i "s/start_motion_daemon=no/start_motion_daemon=yes/" \
/etc/default/motion
```

7) Modify the configuration of **/etc/motion/motion.conf**

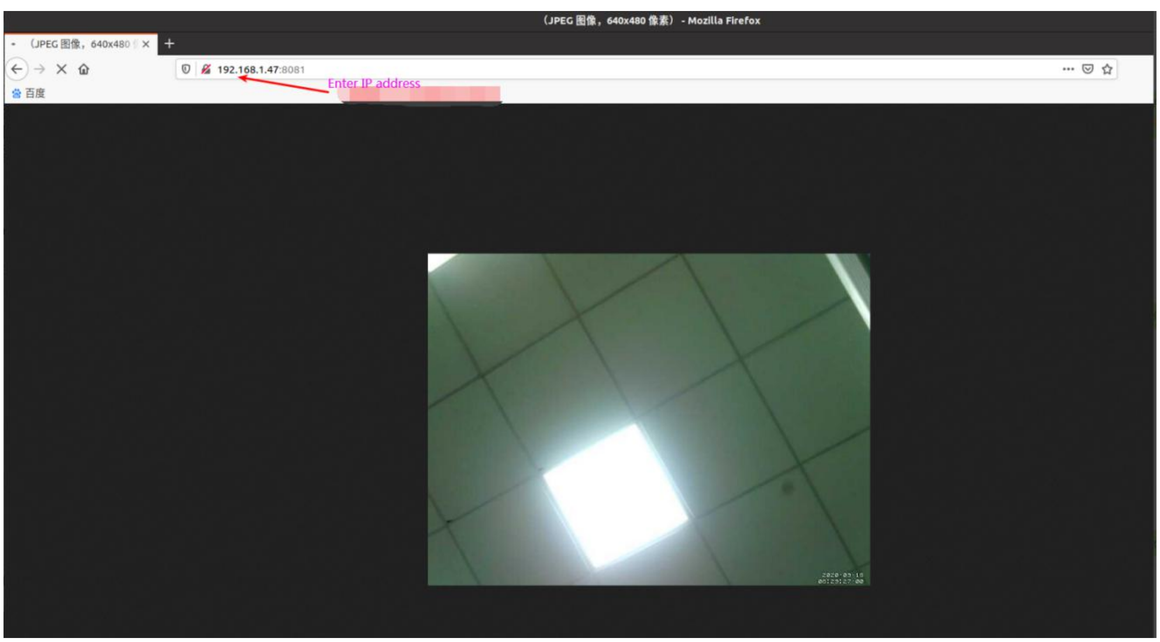
```
root@orangepi:~# sed -i "s/stream_localhost on/stream_localhost off/" \
/etc/motion/motion.conf
```

8) Then restart the motion service

```
root@orangepi:~# /etc/init.d/motion restart
[ ok ] Restarting motion (via systemctl): motion.service.
root@orangepi:~#
```

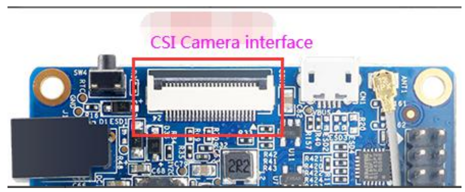
9) Before using motion, please make sure that the Orange Pi development board can connect to the network normally, and then obtain the IP address of the development board through the ifconfig command

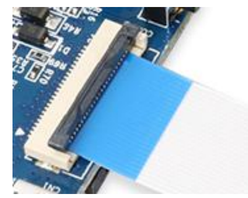
3. 20. 4. Linux3.4 system ov5640 camera test



1) linux5.4 system currently only supports ov5640 camera, not gc2035

2) First connect the Ov5640 camera adapter board to the CIS camera interface of the development board through a cable, and then start the linux system after connecting the camera (don't plug in the camera after powering on)





3) After entering the system, check the loading status of the ov5640 kernel module

```
root@orangepi:~# lsmod | grep "ov5640"
ov5640                    28672   1
v4l2_fwnode               24576   2 ov5640,sun6i_csi
videodev                 151552   7
ov5640,v4l2_fwnode,sunxi_cedrus,videobuf2_common,sun6i_csi,v4l2_mem2mem,video
buf2_v4l2
mc                        36864   7
ov5640,sunxi_cedrus,videobuf2_common,videodev,sun6i_csi,v4l2_mem2mem,videobuf
2_v4l2
```

4) Then use v4l2-ctl (note that l in v4l2 is a lowercase letter l, not a number 1) command to view the device node of the CSI camera. From the output below, we can see that the device node corresponding to the USB camera is **/dev/video0**

```
root@orangepi:~# apt update
root@orangepi:~# apt install v4l-utils
root@orangepi:~# v4l2-ctl   --list-devices
sun6i-csi (platform:camera):
        /dev/video0

cedrus (platform:cedrus):
        /dev/video1
```

5) Then start to install the camera test software motion

```
root@orangepi:~# apt update
root@orangepi:~# apt install motion
```

6) Modify the configuration of /etc/default/motion, change start_motion_daemon=no to start_motion_daemon=yes

```
root@orangepi:~# sed -i "s/start_motion_daemon=no/start_motion_daemon=yes/" \
/etc/default/motion
```

7) Modify the configuration of /etc/motion/motion.conf and set the resolution to 640x480 (other resolutions are not currently supported)

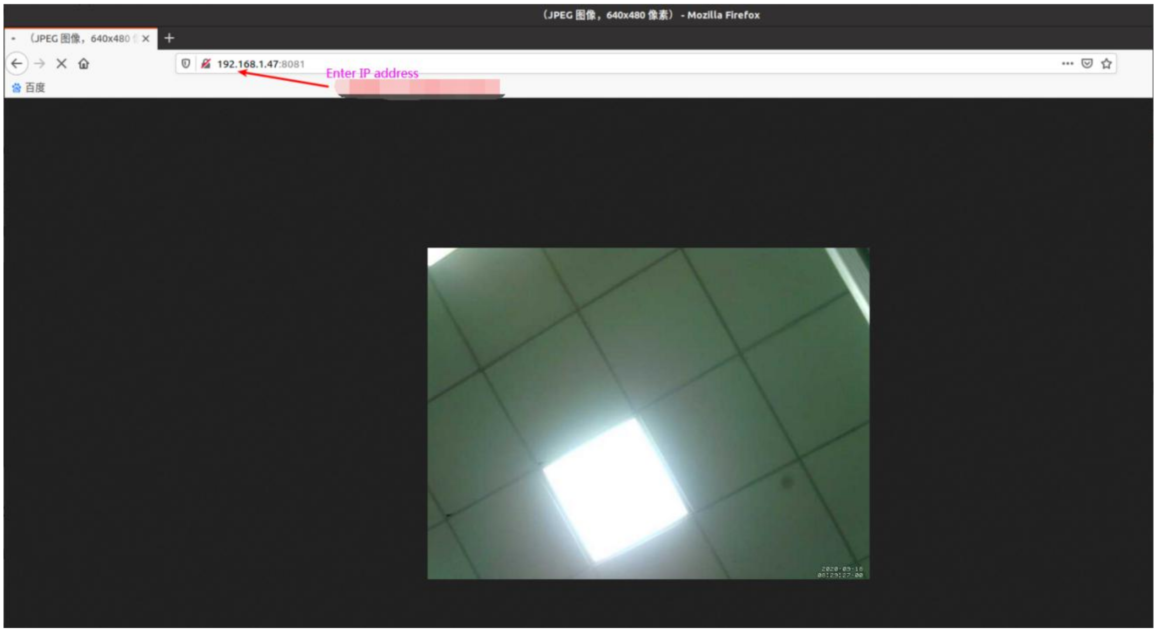
```
root@orangepi:~# sed -i "s/width 320/width 640/" /etc/motion/motion.conf
root@orangepi:~# sed -i "s/height 240/height 480/" /etc/motion/motion.conf
root@orangepi:~# sed -i "s/stream_localhost on/stream_localhost off/" \
/etc/motion/motion.conf
```

8) Then restart the motion service

```
root@orangepi:~# /etc/init.d/motion restart
[ ok ] Restarting motion (via systemctl): motion.service.
root@orangepi:~#
```

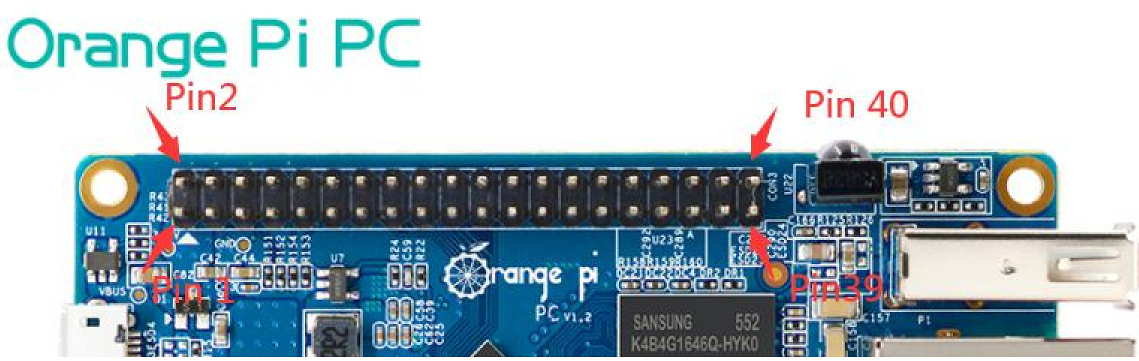
9) Before using motion, please make sure that the Orange Pi development board can connect to the network normally, and then obtain the IP address of the development board through the ifconfig command

10) Then enter the [IP address of the development board: 8081] in the Firefox browser to see the image output by the ov5640



3. 21. 40 Pin interface pin description

1) Please refer to the figure below for the sequence of the 40 pin pins of the Orange Pi PC development board



2) The functions of the 40 pin pins of the Orange Pi PC development board are shown in the table below

GPIO No.	GPIO	Function	Pin	Pin	Function	GPIO	GPIO No.
		3.3v	1	2	5v		
12	PA12	SDA.0	3	4	5v		
11	PA11	SCL.0	5	6	GND		

6	PA6	PA6	7	8	TXD.3	PA13	13
		GND	9	10	RXD.3	PA14	14
1	PA1	RXD.2	11	12	PD14	PD14	110
0	PA0	TXD.2	13	14	GND		
3	PA3	CTS.2	15	16	PC4	PC4	68
		3.3v	17	18	PC7	PC7	71
64	PC0	MOSI.0	19	20	GND		
65	PC1	MISO.0	21	22	RTS.2	PA2	2
66	PC2	SCLK.0	23	24	CE.0	PC3	67
		GND	25	26	PA21	PA21	21
19	PA19	SDA.1	27	28	SCL.1	PA18	18
7	PA7	PA7	29	30	GND		
8	PA8	PA8	31	32	RTS.1	PG8	200
9	PA9	PA9	33	34	GND		
10	PA10	PA10	35	36	CTS.1	PG9	201
20	PA20	PA20	37	38	TXD.1	PG6	198
		GND	39	40	RXD.1	PG7	199

3. 22. Install wiringOP

1) Download the code of wiringOP

```
root@orangepi:~# apt update
root@orangepi:~# apt install git
root@orangepi:~# git clone https://github.com/orangepi-xunlong/wiringOP
```

2) Compile and install wiringOP

```
root@orangepi:~# cd wiringOP
root@orangepi:~/wiringOP# ./build clean
root@orangepi:~/wiringOP# ./build
```

3) The output of the test gpio readall command is as follows

```
root@orangepi:~# gpio readall
 +------+-----+----------+------+---+OrangePiH3+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   12 |   0 |    SDA.0 | ALT2 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   11 |   1 |    SCL.0 | ALT2 | 0 |  5 || 6  |   |      | GND      |     |      |
 |    6 |   2 |      PA6 |  OFF | 0 |  7 || 8  | 0 | ALT3 | TXD.3    | 3   | 13   |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT3 | RXD.3    | 4   | 14   |
 |    1 |   5 |    RXD.2 | ALT2 | 0 | 11 || 12 | 0 | OFF  | PD14     | 6   | 110  |
 |    0 |   7 |    TXD.2 | ALT2 | 0 | 13 || 14 |   |      | GND      |     |      |
 |    3 |   8 |    CTS.2 |  OFF | 0 | 15 || 16 | 0 | OFF  | PC04     | 9   | 68   |
 |      |     |     3.3V |      |   | 17 || 18 | 0 | OFF  | PC07     | 10  | 71   |
 |   64 |  11 |   MOSI.0 | ALT3 | 0 | 19 || 20 |   |      | GND      |     |      |
 |   65 |  12 |   MISO.0 | ALT3 | 0 | 21 || 22 | 0 | OFF  | RTS.2    | 13  | 2    |
 |   66 |  14 |   SCLK.0 | ALT3 | 0 | 23 || 24 | 0 | ALT3 | CE.0     | 15  | 67   |
 |      |     |      GND |      |   | 25 || 26 | 0 | OFF  | PA21     | 16  | 21   |
 |   19 |  17 |    SDA.1 | ALT3 | 0 | 27 || 28 | 0 | ALT3 | SCL.1    | 18  | 18   |
 |    7 |  19 |     PA07 |  OFF | 0 | 29 || 30 |   |      | GND      |     |      |
 |    8 |  20 |     PA08 |  OFF | 0 | 31 || 32 | 0 | OFF  | RTS.1    | 21  | 200  |
 |    9 |  22 |     PA09 |  OFF | 0 | 33 || 34 |   |      | GND      |     |      |
 |   10 |  23 |     PA10 |  OFF | 0 | 35 || 36 | 0 | OFF  | CTS.1    | 24  | 201  |
 |   20 |  25 |     PA20 |  OFF | 0 | 37 || 38 | 0 | ALT2 | TXD.1    | 26  | 198  |
 |      |     |      GND |      |   | 39 || 40 | 0 | ALT2 | RXD.1    | 27  | 199  |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+OrangePiH3+---+------+----------+-----+------+
```

3. 23. 40Pin's GPIO, I2C, UART, SPI test

Wiring OP has been adapted to the Orange Pi development board, using wiringOP can test the functions of GPIO, I2C, UART and SPI

Before starting the test, please make sure that wiringOP has been compiled and installed by referring to the section Installing wiringOP

3. 23. 1. Ordinary GPIO port test

1) Let's take pin 7-corresponding to GPIO as PA6-corresponding to wPi serial number as 2-as an example to demonstrate how to set the high and low levels of GPIO

```
root@orangepi:~# gpio readall
 +------+-----+----------+------+---+-OrangePiH3+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   12 |   0 |    SDA.0 | ALT2 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   11 |   1 |    SCL.0 | ALT2 | 0 |  5 || 6  |   |      | GND      |     |      |
 |    6 |   2 |      PA6 |  OFF | 0 |  7 || 8  | 0 | ALT3 | TXD.3    | 3   | 13   |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT3 | RXD.3    | 4   | 14   |
 |    1 |   5 |    RXD.2 | ALT2 | 0 | 11 || 12 | 0 | OFF  | PD14     | 6   | 110  |
```

2) First set the GPIO port to output mode, the third parameter needs to be the serial number of the wPi corresponding to the input pin

root@orangepi:~# **gpio mode 2 out**

Use gpio readall to see that the Mode of pin 7 is displayed as **OUT**

```
root@orangepi:~# gpio mode 2 out
root@orangepi:~# gpio readall
 +------+-----+----------+------+---+-OrangePiH3+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   12 |   0 |    SDA.0 | ALT2 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   11 |   1 |    SCL.0 | ALT2 | 0 |  5 || 6  |   |      | GND      |     |      |
 |    6 |   2 |      PA6 |  OUT | 0 |  7 || 8  | 0 | ALT3 | TXD.3    | 3   | 13   |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT3 | RXD.3    | 4   | 14   |
```

3) Then set the GPIO port to output low level. After setting, you can use a multimeter to measure the value of the pin voltage. If it is 0v, it means that the low level is set successfully.

root@orangepi:~# **gpio write 2 0**

Use gpio readall to see that the value (V) of pin 7 has become 0

```
root@orangepi:~# gpio write 2 0
root@orangepi:~# gpio readall
 +------+-----+----------+------+---+-OrangePiH3+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   12 |   0 |    SDA.0 | ALT2 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   11 |   1 |    SCL.0 | ALT2 | 0 |  5 || 6  |   |      | GND      |     |      |
 |    6 |   2 |      PA6 |  OUT | 0 |  7 || 8  | 0 | ALT3 | TXD.3    | 3   | 13   |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT3 | RXD.3    | 4   | 14   |
```

4) Then set the GPIO port to output high level. After setting, you can use a multimeter to measure the value of the pin voltage. If it is 3.3v, it means that the high level is set

successfully.

```
root@orangepi:~# gpio write 2 1
```

Use gpio readall to see that the value (V) of pin 7 has become 1

```
root@orangepi:~# gpio write 2 1
root@orangepi:~# gpio readall
 +------+-----+----------+------+---+OrangePiH3+---+------+----------+-----+------+
 | GPIO | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | GPIO |
 +------+-----+----------+------+---+----++----+---+------+----------+-----+------+
 |      |     |     3.3V |      |   |  1 || 2  |   |      | 5V       |     |      |
 |   12 |   0 |    SDA.0 | ALT2 | 0 |  3 || 4  |   |      | 5V       |     |      |
 |   11 |   1 |    SCL.0 | ALT2 | 0 |  5 || 6  |   |      | GND      |     |      |
 |    6 |   2 |      PA6 |  OUT | 1 |  7 || 8  | 0 | ALT3 | TXD.3    | 3   | 13   |
 |      |     |      GND |      |   |  9 || 10 | 0 | ALT3 | RXD.3    | 4   | 14   |
```

5) The setting method of other pins is similar, just modify the serial number of wPi to the serial number corresponding to the pin.

3. 23. 2. SPI interface test

1) The linux5.4 system turns off the spi controller in 40pin by default in the dts. If you need to use spi, you need to open the spi configuration first. The linux3.4 system is enabled by default and no additional configuration is required. The linux5.4 system The opening method of spi is as follows:

a. According to the 40pin schematic diagram, the available spi for Orange Pi Pc is spi0

b. Then set overlays=spi-spidev in /boot/orangepiEnv.txt, set param_spidev_spi_bus=0, where 0 means spi0

```
overlays=spi-spidev
param_spidev_spi_bus=0        #Modify to the corresponding spi bus number supported by the development board
```

c. Then restart the system. When booting, you can see the configuration output of SPI DT overlays in the boot log of u-boot

```
788 bytes read in 8 ms (95.7 KiB/s)
Applying kernel provided DT overlay sun8i-h3-spi-spidev.dtbo
```

```
4185 bytes read in 8 ms (510.7 KiB/s)
Applying kernel provided DT fixup script (sun8i-h3-fixup.scr)
```

d. After the system starts, if you can see the SPI device node under /dev, it means the configuration is correct

```
root@orangepi:~# ls /dev/spi*
/dev/spidev0.0
```

2) Then compile the spidev_test test program

a. The compilation command for linux5.4 system is

```
root@orangepi:~/wiringOP/examples# make spidev_test
[CC] spidev_test.c
[link]
```

b. The compilation command for linux3.4 system is

```
root@orangepi:~/wiringOP/examples# make spidev_test_linux3_4
[CC] spidev_test.c
[link]
```

3) Do not short-circuit the mosi and miso pins of spi first, and the output result of running spidev_test is as follows, you can see that the data sent and received are inconsistent

a. The test commands and results of the linux5.4 system are

```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev0.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF F0 0D  | ......@....▒..................▒.
RX | FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF  | ................................
```

b. The test commands and results of the linux3.4 system are

```
root@orangepi:~/wiringOP/examples# ./spidev_test_linux3_4 -D /dev/spidev0.0
spi mode: 0
bits per word: 8
max speed: 500000 Hz (500 KHz)
```

```
00 00 00 00 00 00
00 00 00 00 00 00
00 00 00 00 00 00
00 00 00 00 00 00
00 00 00 00 00 00
00 00 00 00 00 00
00 00
```

4) Then use the Dupont wire to short-circuit the two pins of spi's mosi (corresponding to pin 19) and miso (corresponding to pin 21). The output of the retest is as follows. You can see that the data sent and received are the same, indicating that the spi can be normal use

a. The test commands and results of the Linux5.4 system are

```
root@orangepi:~/wiringOP/examples# ./spidev_test -v -D /dev/spidev0.0
spi mode: 0x0
bits per word: 8
max speed: 500000 Hz (500 KHz)
TX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF F0 0D  | ......@....▒..................▒.
RX | FF FF FF FF FF FF 40 00 00 00 00 95 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF F0 0D  | ......@....▒..................▒.
```

b. The test commands and results of the linux3.4 system are

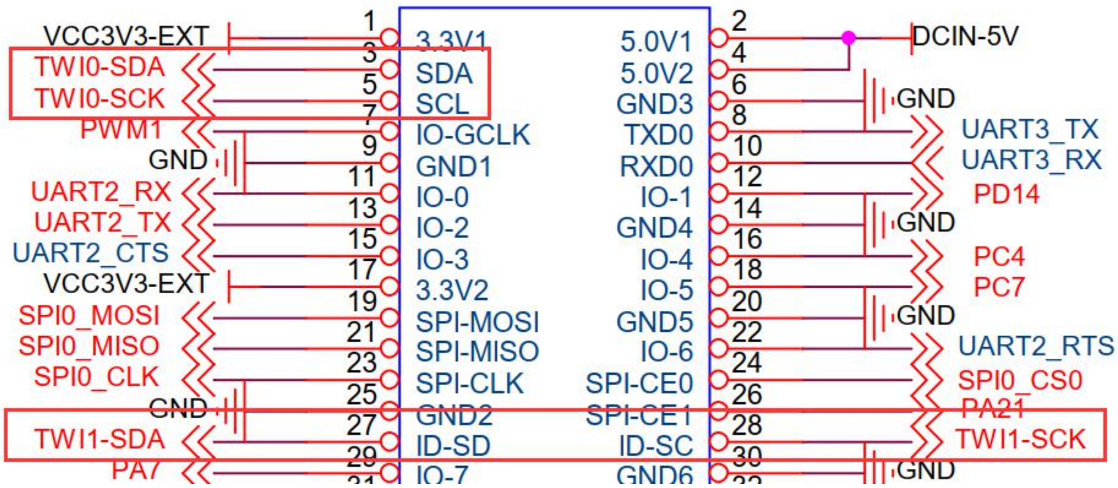
```
root@orangepi:~/wiringOP/examples# ./spidev_test_linux3_4 -D /dev/spidev0.0
spi mode: 0
bits per word: 8
max speed: 500000 Hz (500 KHz)

FF FF FF FF FF FF
40 00 00 00 00 95
FF FF FF FF FF FF
FF FF FF FF FF FF
FF FF FF FF FF FF
DE AD BE EF BA AD
F0 0D
```

3. 23. 3. I2C test

1) The linux5.4 system turns off the i2c controller in 40pin by default in the dts. If you need to use i2c, you need to open the i2c configuration first. The linux3.4 system is all on by default and no additional configuration is required. The linux5.4 system The opening method of i2c is as follows

a. According to the 40pin schematic diagram, the i2c available for Orange Pi Pc are i2c0 and i2c1



b. Then set overlays=i2c0 i2c1 in /boot/orangepiEnv.txt to open the configuration of i2c0 and i2c1 at the same time

```
overlays=i2c0 i2c1
```

c. Then restart the system. When starting, you can see the configuration output of I2C DT overlays in the u-boot startup log

```
Found mainline kernel configuration
29940 bytes read in 6 ms (4.8 MiB/s)
374 bytes read in 8 ms (44.9 KiB/s)
Applying kernel provided DT overlay sun8i-h3-i2c0.dtbo
374 bytes read in 8 ms (44.9 KiB/s)
Applying kernel provided DT overlay sun8i-h3-i2c1.dtbo
```

d. After the system is started, if there are two more i2c device nodes under /dev, the configuration is correct

```
root@orangepi:~# ls /dev/i2c*
```

```
/dev/i2c-0   /dev/i2c-1   /dev/i2c-2   /dev/i2c-3   /dev/i2c-4
```

e. The corresponding relationship of different i2c device nodes is shown below, where

a) I2c0 in 40pin corresponds to /dev/i2c-0

b) I2c1 in 40pin corresponds to /dev/i2c-1

```
root@orangepipcplus:~# ls /sys/class/i2c-adapter/ -lh
total 0
lrwxrwxrwx 1 root root 0 Oct 13 10:46 i2c-0 -> ../../devices/platform/soc/1c2ac00.i2c/i2c-0
lrwxrwxrwx 1 root root 0 Oct 13 10:46 i2c-1 -> ../../devices/platform/soc/1c2b000.i2c/i2c-1
lrwxrwxrwx 1 root root 0 Oct 13 10:46 i2c-2 -> ../../devices/platform/soc/1c2b400.i2c/i2c-2
lrwxrwxrwx 1 root root 0 Oct 13 10:46 i2c-3 -> ../../devices/platform/soc/1f02400.i2c/i2c-3
lrwxrwxrwx 1 root root 0 Oct 13 10:46 i2c-4 -> ../../devices/platform/soc/1ee0000.hdmi/i2c-4
```

2) Then start testing i2c, first install i2c-tools

```
root@orangepi:~# apt update
root@orangepi:~# apt install i2c-tools
```

3) Then connect an i2c device to the 40pin i2c0 or i2c1

	i2c0	i2c1
Sda Pin	Corresponding to pin 3	Corresponding to pin 27
Sck Pin	Corresponding to pin 5	Corresponding to pin 28
Vcc Pin	Corresponding to pin 1	Corresponding to pin 17
Gnd Pin	Corresponding to pin 6	Corresponding to pin 25

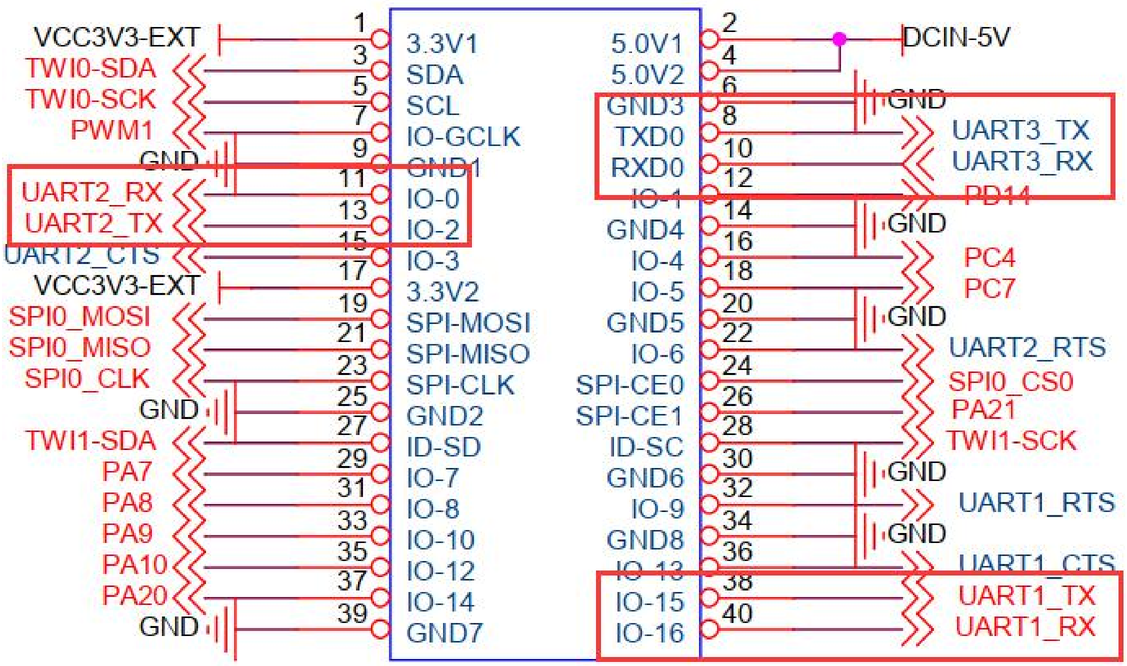
4) Then use i2cdetect -y 0 (where 0 means i2c0, i2c1 needs to be modified to i2cdetect -y 1) command. If the address of the connected i2c device can be detected, it means that i2c can be used normally

```
root@orangepi:~# i2cdetect -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- 57 -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
root@orangepi:~#
```

3. 23. 4. UART test

1) The linux5.4 system turns off the uart controller in the 40pin by default in the dts. If you need to use uart, you need to open the uart configuration first. The linux3.4 system is enabled by default and no additional configuration is required. The linux5.4 system The opening method of uart is as follows:

a. According to the 40pin schematic diagram, the uart available for Orange Pi Pc are uart1, uart2 and uart3



b. Then set overlays=uart1 uart2 uart3 in /boot/orangepiEnv.txt to open the configuration of uart1, uart2 and uart3 at the same time

```
overlays=uart1 uart2 uart3
```

c. Then restart the system. When booting, you can see the configuration output of UART DT overlays in the boot log of u-boot

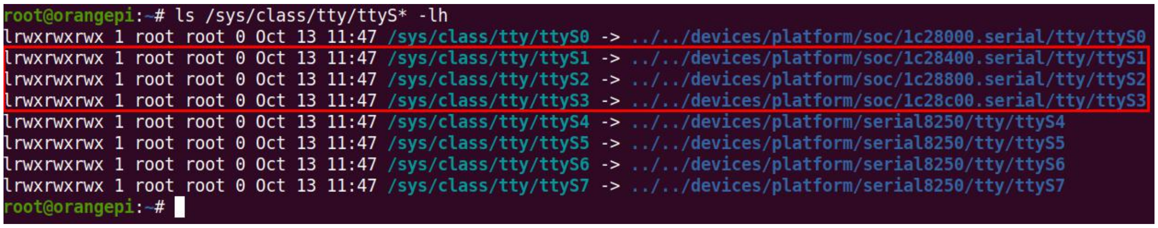
```
Applying kernel provided DT overlay sun8i-h3-uart1.dtbo
502 bytes read in 10 ms (48.8 KiB/s)
Applying kernel provided DT overlay sun8i-h3-uart2.dtbo
502 bytes read in 5 ms (97.7 KiB/s)
Applying kernel provided DT overlay sun8i-h3-uart3.dtbo
4155 bytes read in 4 ms (1013.7 KiB/s)
Applying kernel provided DT fixup script (sun8i-h3-fixup.scr)
```

d. After the system is started, you can see the information of ttyS1, ttyS2 and ttyS3 under /sys/class/tty, where

a) uart1 in 40pin corresponds to /dev/ttyS1

b) uart2 in 40pin corresponds to /dev/ttyS2

c) uart3 in 40pin corresponds to /dev/ttyS3



```
root@orangepi:~# ls /sys/class/tty/ttyS* -lh
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS0 -> ../../devices/platform/soc/1c28000.serial/tty/ttyS0
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS1 -> ../../devices/platform/soc/1c28400.serial/tty/ttyS1
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS2 -> ../../devices/platform/soc/1c28800.serial/tty/ttyS2
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS3 -> ../../devices/platform/soc/1c28c00.serial/tty/ttyS3
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS4 -> ../../devices/platform/serial8250/tty/ttyS4
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS5 -> ../../devices/platform/serial8250/tty/ttyS5
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS6 -> ../../devices/platform/serial8250/tty/ttyS6
lrwxrwxrwx 1 root root 0 Oct 13 11:47 /sys/class/tty/ttyS7 -> ../../devices/platform/serial8250/tty/ttyS7
root@orangepi:~#
```

2) Then start to test the uart interface, first use the Dupont line to short-circuit the rx and tx of the uart interface to be tested

	uart1	uart2	uart3
Tx Pin	Corresponding to pin 38	Corresponding to pin 13	Corresponding to pin 8
Rx Pin	Corresponding to pin 40	Corresponding to pin 11	Corresponding to pin 10

3) Then modify the serial test program serialTest in wiringOP

```
int main ()
{
  int fd ;
  int count ;
  unsigned int nextTime ;

  if ((fd = serialOpen ("/dev/ttyS2", 115200)) < 0)
  {
    fprintf (stderr, "Unable to open serial device: %s\n", strerror (errno)) ;
    return 1 ;
```


4) Recompile the serial test program serialTest in wiringOP

```
root@orangepi:~/wiringOP/examples# make serialTest
[CC] serialTest.c
[link]
root@orangepi:~/wiringOP/examples#
```

5) Finally run the serialTest, if you can see the following print, it means that the serial communication is normal

```
root@orangepi:~/wiringOP/examples# ./serialTest
```

```
Out:    0:   ->    0
Out:    1:   ->    1
Out:    2:   ->    2
Out:    3:   ->    3
Out:    4:   ->    4
Out:    5:   ->    5
Out:    6:   ->    6
Out:    7:   ->    7
Out:    8:   ->    8^C
```

3. 24. How to use 0.96 inch OLED module with I2C interface

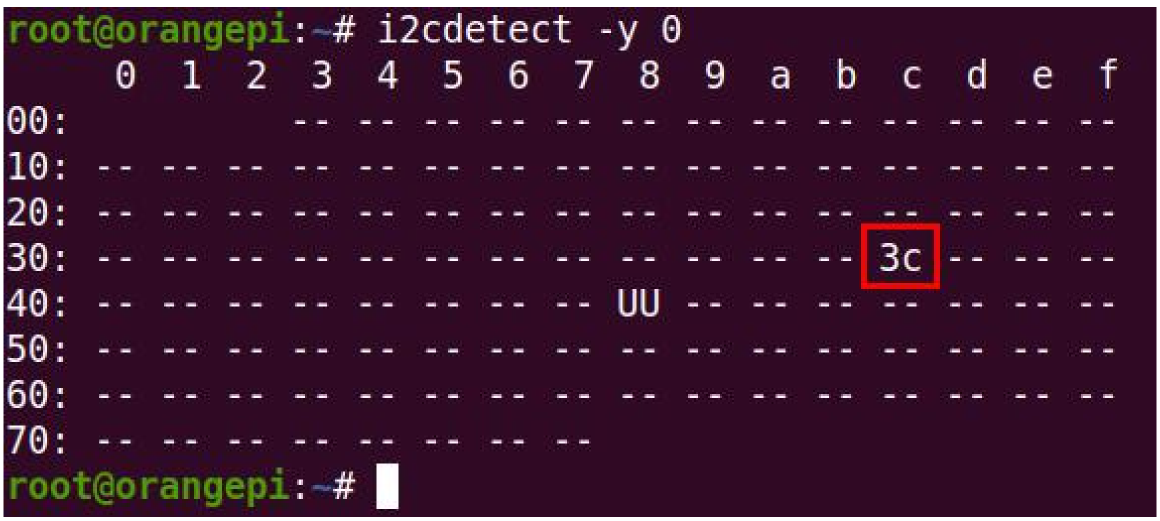
1) The 0.96 inch OLED module of Orange Pi is shown in the figure below, and its 7-bit i2c slave address is 0x3c

2) First connect the 0.96 inch OLED module to the 40pin interface of the Orange Pi development board through the DuPont cable. The wiring method is as follows

Pins of OLED module	Description	Development board 40pin interface corresponding pin
GND	Power ground	6 Pin
VCC	5V	2 Pin
SCL	I2C clock line	5 Pin
SDA	I2C data cable	3 Pin
RST	Connect to 3.3V	1 Pin
DC	Connect to GND	9 Pin
CS	Connect to GND	25 Pin

3) After connecting the OLED module to the development board, first use the i2c-tools tool to check whether the address of the OLED module can be scanned

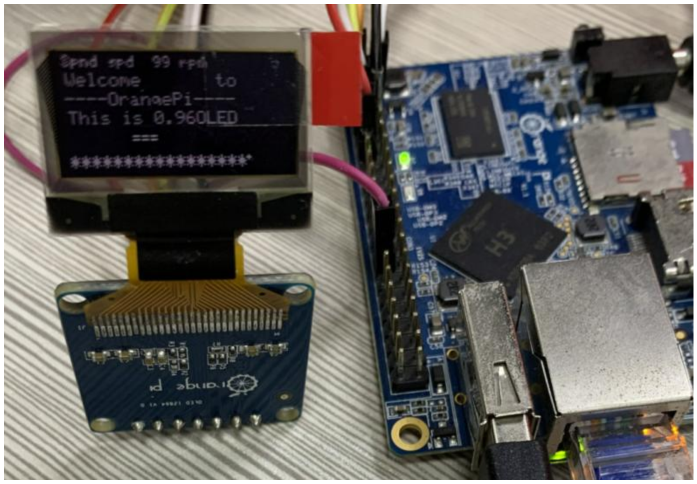
```
root@orangepi:~# apt update
root@orangepi:~# apt install i2c-tools
root@orangepi:~# i2cdetect -y 0
```



4) Then you can use the oled_demo in wiringOP to test the OLED module, the test steps are as follows

```
root@orangepi:~# git clone https://github.com/orangepi-xunlong/wiringOP
root@orangepi:~# cd wiringOP
root@orangepi:~/wiringOP# ./build clean && ./build
root@orangepi:~/wiringOP# cd examples
root@orangepi:~/wiringOP/examples# make oled_demo
root@orangepi:~/wiringOP/examples# ./oled_demo /dev/i2c-0
---------start--------
----------end---------
```

5) After running oled_demo, you can see the following output on the OLED screen



3. 25. How to use SPI LCD display

Note: This method is only applicable to linux3.4 kernel systems, and linux5.4 kernel systems cannot be used

3. 25. 1. 2.4 inch SPI LCD display

1) The link to the tested LCD display details page is as follows

http://www.lcdwiki.com/2.4inch_SPI_Module_ILI9341_SKU:MSP2402

2) The wiring method of the LCD display and the development board is as follows

TFT SPI module pins	The corresponding pins of development board 40pin	GPIO -- GPIO num
VCC-5V	2 Pin	
GND	6 Pin	
CS	24 Pin	
RESET	12 Pin	PD14 -- 110
D/C	16 Pin	PC4 -- 68
SDI(MOSI)	19 Pin	
SCK	23 Pin	
LED	1 Pin	
SDO(MISO)	21 Pin	

3) After connecting the display to the development board, use the following command to

load the **fbtft_device** kernel module

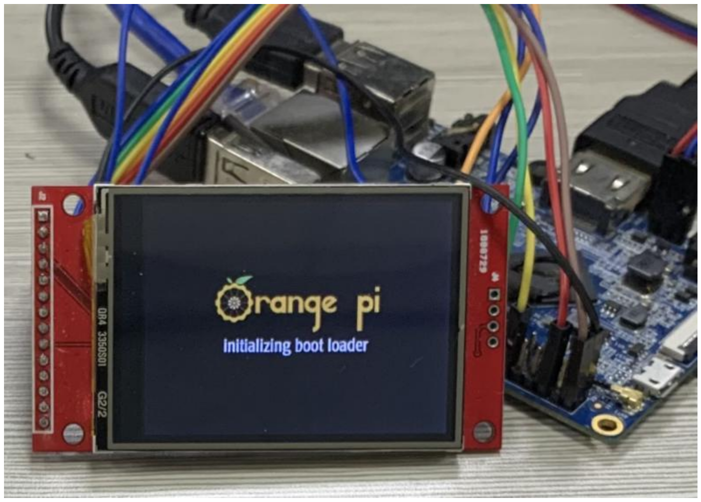
```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9341   busnum=0 cs=0
gpios=reset:110,dc:68 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

4) When the **fbtft_device** kernel module is loaded, the correct output log of the dmesg command is shown below, and the log can know that the framebuffer used by the LCD display is **fb8**

```
root@orangepi:~# dmesg | tail
root@orangepipc:~# dmesg | tail
[    82.034708] fbtft_device:  SPI devices registered:
[    82.034751] fbtft_device:      spidev spi0.0 33000kHz 8 bits mode=0x00
[    82.034779] fbtft_device:  'fb' Platform devices registered:
[    82.034931] fbtft_device: Deleting spi0.0
[    82.036030] fbtft_device:  GPIOS used by 'fb_ili9341':
[    82.036054] fbtft_device:    'reset' = GPIO110
[    82.036072] fbtft_device:    'dc' = GPIO68
[    82.036088] fbtft_device:  SPI devices registered:
[    82.036117] fbtft_device:      fb_ili9341 spi0.0 65000kHz 8 bits mode=0x00
[    82.365862] graphics fb8: fb_ili9341 frame buffer, 320x240, 150 KiB video memory,
64 KiB buffer memory, fps=20, spi0.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo picture on the LCD display

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



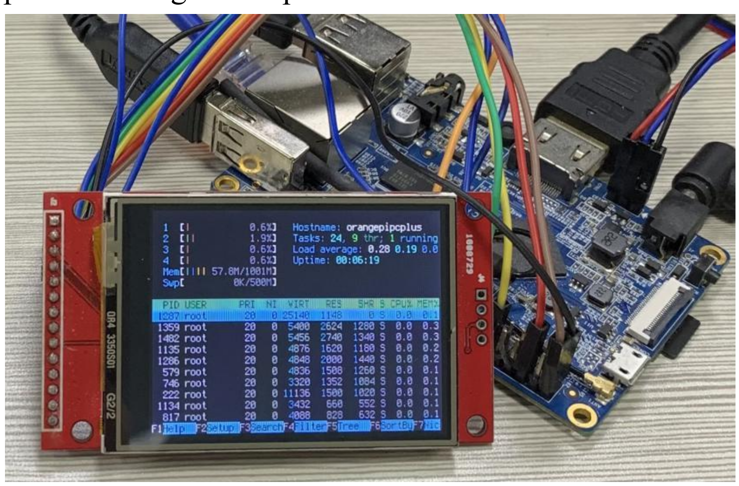
6) You can also map the output of tty1 to the fb device of the LCD display-**fb8**. After the mapping is completed, HDMI will no longer have image output.

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display, please use the following command

```
root@orangepi:~# con2fbmap 1 0
```

Below is the output of running the htop command



7) Because the default terminal font is too large, the screen cannot display too much content, you can use the following method to reduce the terminal font

a. Run first **dpkg-reconfigure console-setup**

```
root@orangepi:~# apt-get update
```

```
root@orangepi:~# apt-get install kbd
root@orangepi:~# dpkg-reconfigure console-setup
```

b. Terminal coding selection **UTF-8**

```
KOI8-U
TIS-620
UTF-8
VISCII

<Ok>          <Cancel>
```

c. Then choose **Guess optimal character set**

```
. Combined - Latin; Slavic Cyrillic; Greek
. Combined - Latin; Slavic and non-Slavic Cyrillic
Guess optimal character set

        <Ok>                                   <Cancel>
```

d. Then choose **Terminus**

```
Fixed
Goha
GohaClassic
Terminus
TerminusBold
TerminusBoldVGA
VGA
Do not change the boot/kernel font
Let the system select a suitable font

<Ok>                                   <Cancel>
```

e. Finally select the font size as 6x12

```
6x12 (framebuffer only)
8x14
8x16
10x20 (framebuffer only)
11x22 (framebuffer only)
12x24 (framebuffer only)
14x28 (framebuffer only)
16x32 (framebuffer only)

<Ok>                                   <Cancel>
```

f. After setting, you can see that the font on the LCD display becomes smaller

8) Method for setting system startup to automatically load fbtft_device module

a. Create a new **/etc/modules-load.d/fbtft.con** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options  fbtft_device custom name=fb_ili9341  busnum=0 cs=0
gpios=reset:110,dc:68 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system and you can see that the kernel modules related to fbtft_device have been automatically loaded

9) If you want the linux system to automatically map the console to the LCD display after booting, please add the following configuration to **/boot/orangepiEnv.txt,** and then restart the system to see the LCD display output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=fbcon=map:8
```

3. 25. 2. 3.2 inch RPi SPI LCD display

1) The link to the tested LCD display details page is as follows

```
http://www.lcdwiki.com/3.2inch_RPi_Display
```

2) The wiring method of the LCD display and the development board is as follows

3) After connecting the LCD display to the development board, use the following command to load the **fbtft_device** kernel module

```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9341  busnum=0 cs=0
```

```
gpios=reset:0,dc:3 rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

4) When the **fbtft_device** kernel module is loaded, the correct output log of the dmesg command is shown below, and the log can know that the framebuffer used by the LCD screen is **fb8**

```
root@orangepipc:~# dmesg | tail
[    99.471345] fbtft_device:   SPI devices registered:
[    99.471383] fbtft_device:       spidev spi0.0 33000kHz 8 bits mode=0x00
[    99.471405] fbtft_device:   'fb' Platform devices registered:
[    99.471554] fbtft_device: Deleting spi0.0
[    99.472469] fbtft_device:   GPIOS used by 'fb_ili9341':
[    99.472493] fbtft_device:     'reset' = GPIO0
[    99.472510] fbtft_device:     'dc' = GPIO3
[    99.472525] fbtft_device:   SPI devices registered:
[    99.472554] fbtft_device:       fb_ili9341 spi0.0 65000kHz 8 bits mode=0x00
[    99.796157] graphics fb8: fb_ili9341 frame buffer, 320x240, 150 KiB video memory, 64 KiB buffer memory, fps=20, spi0.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo picture on the LCD screen

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```

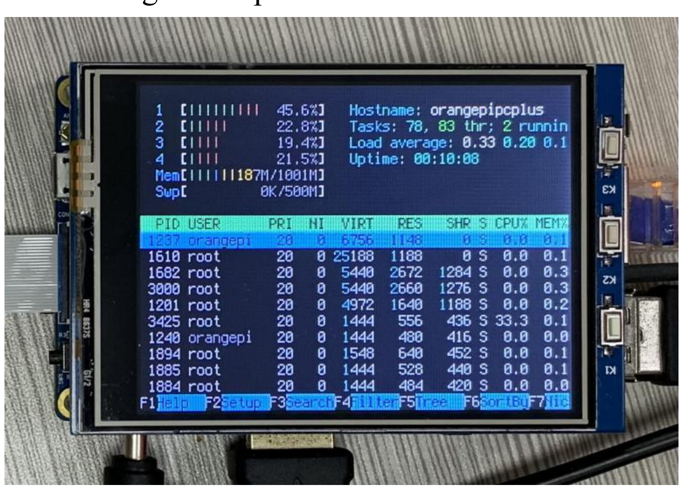
6) You can also map the output of tty1 to the fb device of the LCD screen-**fb8**. After the mapping is completed, HDMI will no longer have image output.

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display, please use the following command

```
root@orangepi:~# con2fbmap 1 0
```

Below is the output of running the htop command

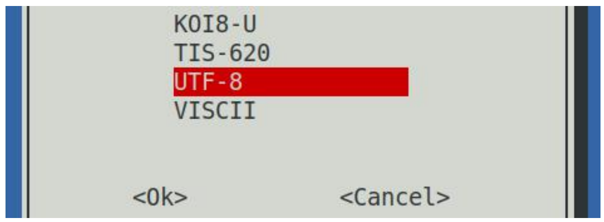


7) Because the default terminal font is too large, the screen cannot display too much content, you can use the following method to reduce the terminal font

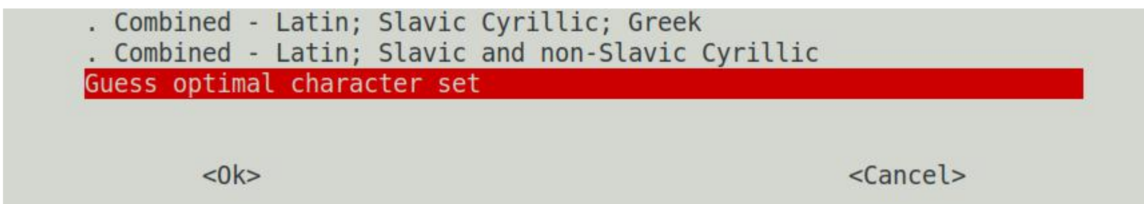
a. Run first **dpkg-reconfigure console-setup**

```
root@orangepi:~# apt-get update
root@orangepi:~# apt-get install kbd
root@orangepi:~# dpkg-reconfigure console-setup
```

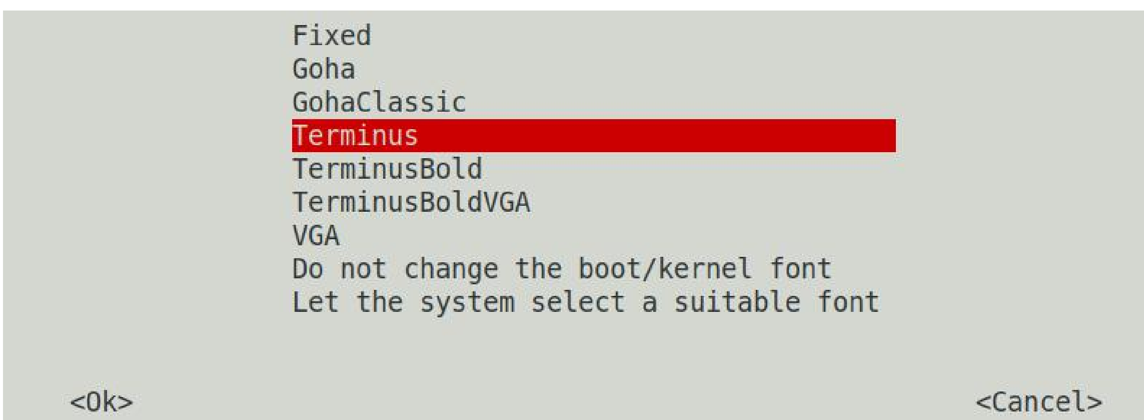
b. Terminal coding selection **UTF-8**



c. Then choose **Guess optimal character set**



d. Then choose **Terminus**



e. Finally select the font size as 6x12

```
6x12 (framebuffer only)
8x14
8x16
10x20 (framebuffer only)
11x22 (framebuffer only)
12x24 (framebuffer only)
14x28 (framebuffer only)
16x32 (framebuffer only)

<Ok>                                              <Cancel>
```

f. After setting, you can see that the font on the LCD screen becomes smaller

8) Method for setting system startup to automatically load fbtft_device module

a. Create a new **/etc/modules-load.d/fbtft.con** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modules-load.d/fbtft.conf
fbtft_device
```

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

```
root@orangepi:~# cat /etc/modprobe.d/fbtft.conf
options fbtft_device custom name=fb_ili9341    busnum=0 cs=0    gpios=reset:0,dc:3
rotate=90 speed=65000000 bgr=1 txbuflen=65536
```

c. Then restart the linux system and you can see that the kernel modules related to fbtft_device have been automatically loaded

9) If you want the linux system to automatically map the console to the LCD screen after booting, please add the following configuration in **/boot/orangepiEnv.txt,** and then restart the system to see the LCD screen output

```
root@orangepi:~# cat /boot/orangepiEnv.txt | grep "fbcon"
extraargs=fbcon=map:8
```

3. 25. 3. 3.5 inch SPI LCD display

1) The link to the details page of the tested LCD display is as follows

http://www.lcdwiki.com/3.5inch_SPI_Module_ILI9488_SKU:MSP3520

2) The wiring method of the LCD display and the development board is as follows

TFT SPI module pins	The corresponding pins of development board 40pin	GPIO -- GPIO num
VCC	1 Pin	
GND	6 Pin	
CS	24 Pin	
RESET	12 Pin	PD14 -- 110
DC/RS	16 Pin	PC04 -- 68
SDI(MOSI)	19 Pin	
SCK	23 Pin	
LED	18 Pin	PC7 -- 71
SDO(MISO)	21 Pin	

3) After connecting the display to the development board, use the following command to load the fbtft_device kernel module

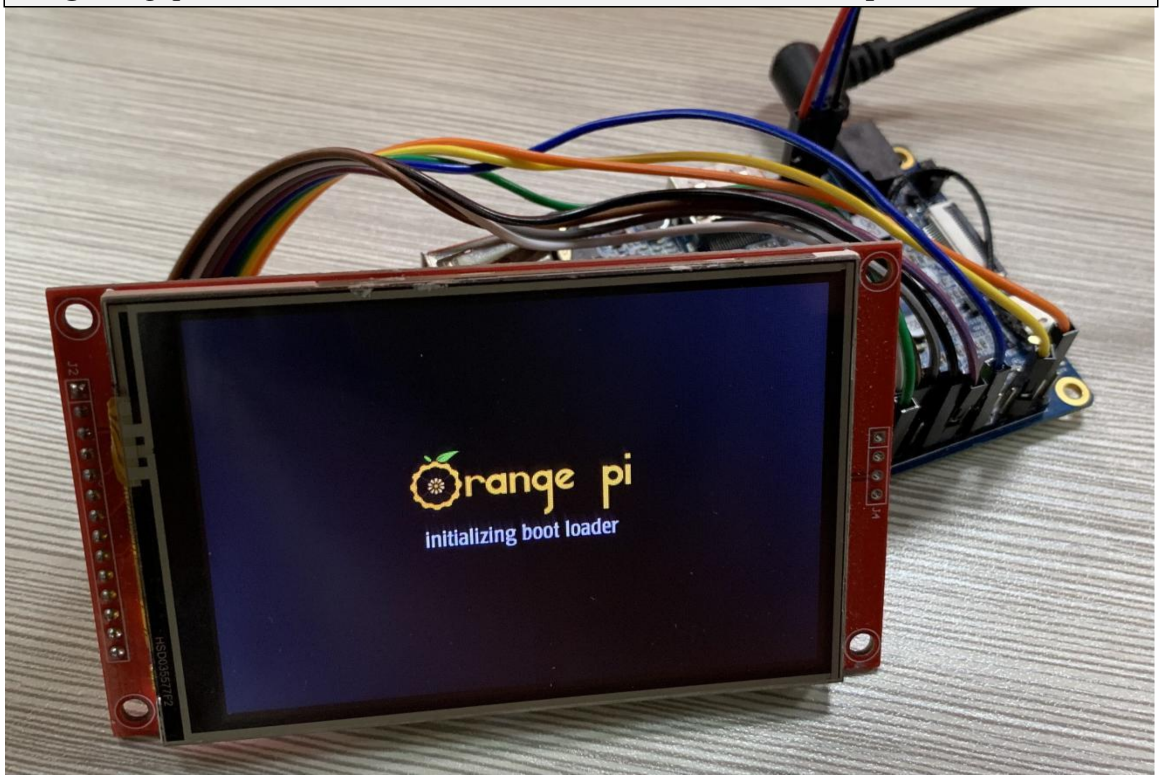
```
root@orangepi:~# modprobe fbtft_device custom name=fb_ili9488  busnum=0 cs=0
gpios=reset:110,dc:68,led:71 rotate=270 speed=65000000 bgr=1 txbuflen=65536
```

4) When the **fbtft_device** kernel module is loaded, the correct output log of the dmesg command is shown below, and the log can know that the framebuffer used by the LCD display is **fb8**

```
root@orangepipc:~# dmesg | tail
[   273.581459] fbtft_device:        spidev spi0.0 33000kHz 8 bits mode=0x00
[   273.581483] fbtft_device:   'fb' Platform devices registered:
[   273.581628] fbtft_device: Deleting spi0.0
[   273.582486] fbtft_device:   GPIOS used by 'fb_ili9488':
[   273.582509] fbtft_device:      'reset' = GPIO110
[   273.582526] fbtft_device:      'dc' = GPIO68
[   273.582543] fbtft_device:      'led' = GPIO71
[   273.582563] fbtft_device:   SPI devices registered:
[   273.582598] fbtft_device:        fb_ili9488 spi0.0 65000kHz 8 bits mode=0x00
[   273.955952] graphics fb8: fb_ili9488 frame buffer, 480x320, 300 KiB video memory,
64 KiB buffer memory, fps=100, spi0.0 at 65 MHz
```

5) Then use the following command to display the Orange Pi logo picture on the LCD display

```
root@orangepi:~# apt update
root@orangepi:~# apt -y install fbi
root@orangepi:~# fbi -vt 1 -noverbose -d /dev/fb8 /boot/boot.bmp
```



6) You can also map the output of tty1 to the fb device of the LCD display-**fb8**. After the mapping is completed, the LCD screen will display the output of the terminal, and HDMI will no longer have image output.

```
root@orangepi:~# con2fbmap 1 8
```

If you want to switch back to HDMI display, please use the following command

root@orangepi:~# **con2fbmap 1 0**

7) Set the method to automatically load the fbtft_device module at system startup

a. Create a new **/etc/modules-load.d/fbtft.con** configuration file, the content of the file is as follows

root@orangepi:~# **cat /etc/modules-load.d/fbtft.conf**
fbtft_device

b. Create a new **/etc/modprobe.d/fbtft.conf** configuration file, the content of the file is as follows

root@orangepi:~# **cat /etc/modprobe.d/fbtft.conf**
options fbtft_device custom name=fb_ili9488 busnum=0 cs=0
gpios=reset:110,dc:68,led:71 rotate=270 speed=65000000 bgr=1 txbuflen=65536

c. Then restart the linux system and you can see that the kernel modules related to fbtft_device have been automatically loaded

8) If you want the linux system to automatically map the console to the LCD display after booting, please add the following configuration to **/boot/orangepiEnv.txt,** and then restart the system to see the LCD display output

root@orangepi:~# **cat /boot/orangepiEnv.txt | grep "fbcon"**

```
extraargs=fbcon=map:8                //Server version system needs to add configuration
extraargs=cma=96M fbcon=map:8        //Configurations that need to be added to the desktop version
```

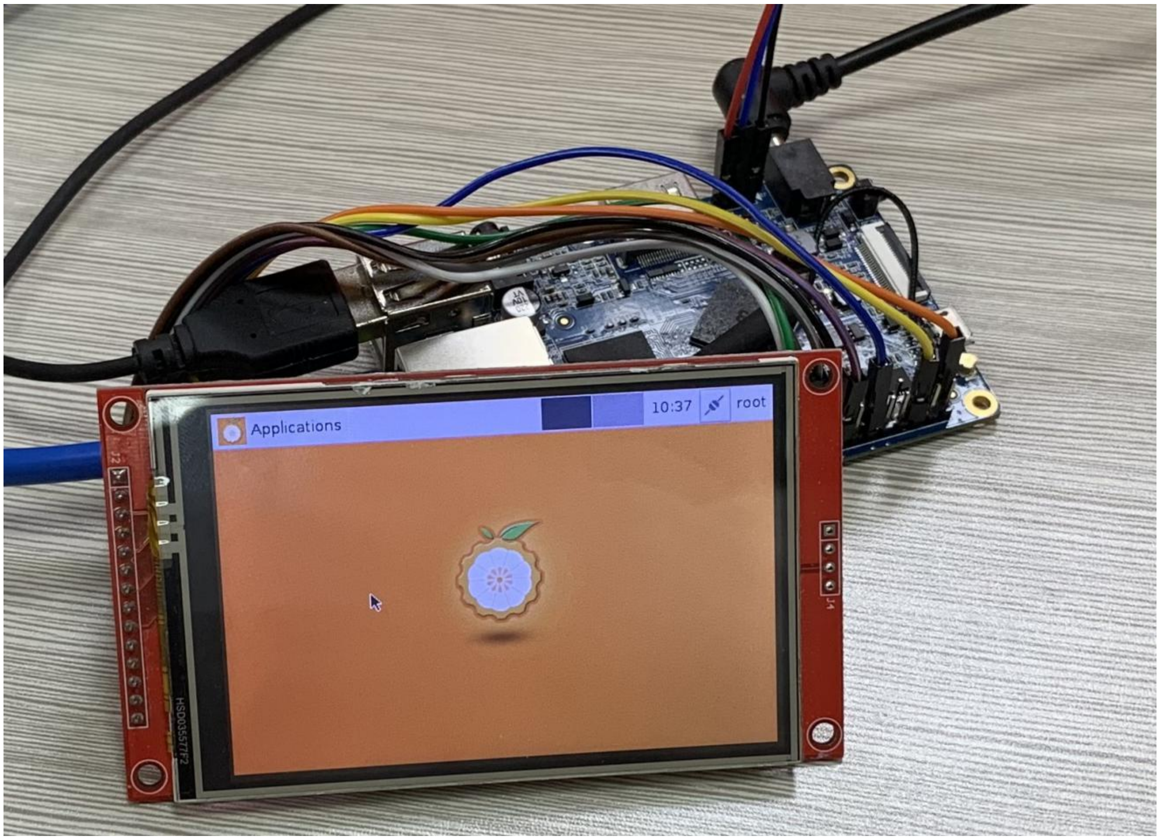
9) If you need to display the desktop version of the system to the LCD screen, first modify the following configuration file, change fb0 to **fb8**, and you can see the desktop displayed on the LCD screen after restarting

```
root@orangepipc:~# cat /etc/X11/xorg.conf.d/50-fbturbo.conf
Section "Device"
        Identifier      "Allwinner A10/A13 FBDEV"
        Driver          "fbturbo"
        Option          "fbdev" "/dev/fb8"

        Option          "SwapbuffersWait" "true"
EndSection
```

10) If you do not restart the system, you can execute the following command, after a few seconds, the LCD screen can also see the desktop of the linux system

```
root@orangepi:~# FRAMEBUFFER=/dev/fb8 startx
```



3. 26. linux3.4 desktop version system GPU driver test method

1) First install **glmark2-es2**

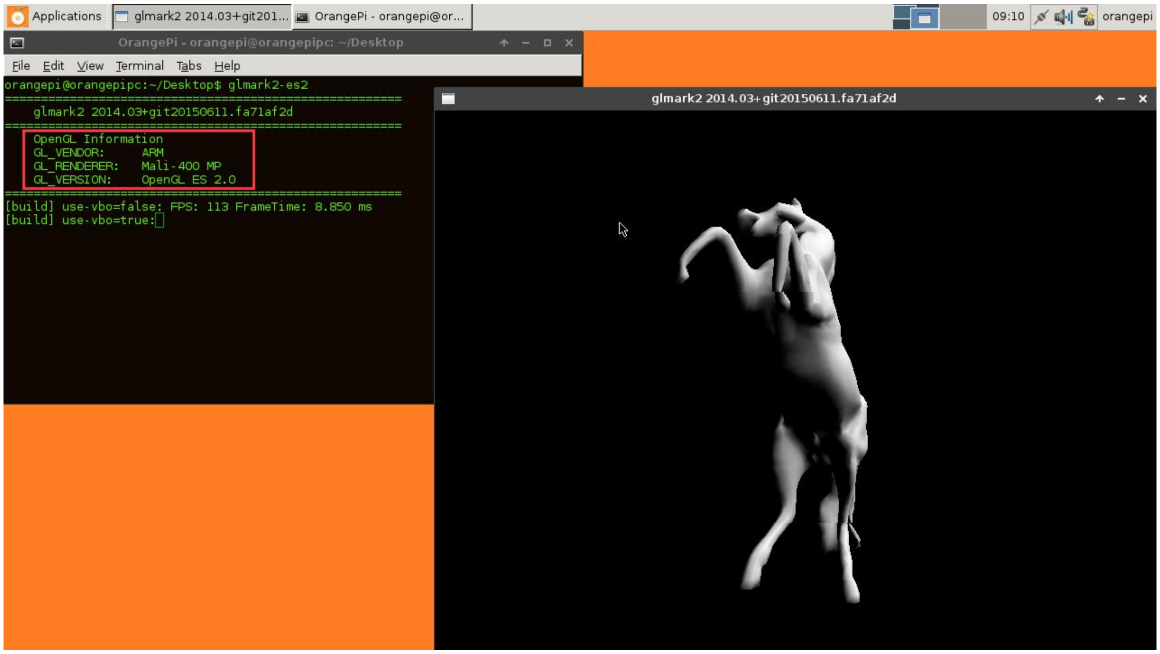
```
root@orangepi:~# apt update
root@orangepi:~# apt install glmark2-es2
```

2) Then enter the desktop of the linux system through the HDMI display, **do not use ssh to log in remotely or serial port to log in to the linux system**

3) Run glmark2-es2

```
root@orangepi:~# glmark2-es2
```

4) It can be seen that OpenGL uses Mali-400 MP, indicating that the GPU can be used normally



3. 27. View the chipid of the H3 chip

Note: This method is only suitable for linux3.4 system, linux5.4 system cannot read

1) The command to view the chipid of the h3 chip is as follows, the chipid of each chip is different, so you can use chipid to distinguish multiple development boards

```
root@orangepi:~# cat /sys/class/sunxi_info/sys_info | grep "chipid"
sunxi_chipid          : 541c035348a0471c0000115000000000
```

3. 28. How to use linux3.4 system TV-OUT

Note: TV-OUT cannot be used in linux5.4 system, the following method is only applicable to linux3.4 system

1) First set the scrpit.bin used by the linux kernel to **orangepipc-tvout.bin**

```
root@orangepi:~# rm /boot/script.bin
root@orangepi:~# ln -sv /boot/bin/orangepipc-tvout.bin /boot/script.bin
'/boot/script.bin' -> '/boot/bin/orangepipc-tvout.bin'
```

2) Then you need to restart the system for the configuration of **orangepipc-tvout.bin** to

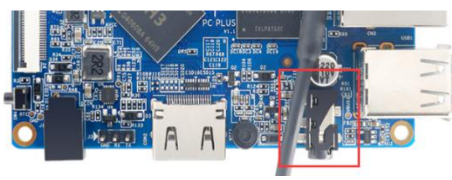
take effect

3) Then use the AV cable to connect the development board to the TV, the steps are as follows

a. First, you need to prepare an AV cable, as shown in the figure below

b. The connection of the AV cable is as follows

a) Black: plug into the analog audio and video interface of the development board



b) Red: plug into the AV video input of the TV

c) Yellow: plug into the right channel audio input of the TV

d) White: plug into the left channel audio input of the TV

4) Then the signal source at one end of the TV needs to select the video input. Different TV setting methods may be different. Please refer to the corresponding instruction manual. If the connection settings are correct, you can see the output of the linux system

TV-OUT on the TV. Up

3. 29. Boot and shutdown method

1) Shut down using the poweroff command

```
root@orangepi:~# poweroff
```

2) You can also short press the power button on the development board to shut down (The **linux3.4 server version is temporarily unavailable)**

3) After shutting down, you need to unplug and plug the power again to boot up

4) The command to restart the linux system is

```
root@orangepi:~# reboot
```

4. Android system instructions

4. 1. Supported Android version

Android version	Kernel version
Android 4.4	linux3.4
Android 7.0	linux4.4

4. 2. Android 4.4 function adaptation situation

Function	Status
HDMI video	OK
HDMI audio	OK
USB2.0 x 3	OK
TF card boot	OK
Network card	OK
Infrared	OK
CVBS audio	OK
CVBS video	OK
MIC recording	OK
USB camera	OK
OV5640 camera	OK
GC2035 camera	OK
button	OK
LED lights	OK
Temperature Sensor	OK
ADB debugging	OK
Mali GPU	OK
Video codec	OK

4. 3. Android 7.0 function adaptation situation

Function	Status
HDMI video	OK
HDMI audio	OK
USB2.0 x 3	OK
TF card boot	OK
Network card	OK
Infrared	OK
CVBS audio	OK
CVBS video	OK
MIC recording	OK
USB camera	OK
OV5640 camera	NO
GC2035 camera	NO
Button	OK
LED lights	OK
Temperature Sensor	OK
ADB debugging	OK
Mali GPU	OK
Video codec	OK

4. 4. Onboard LED light display description

1) LED light display

	Green light	Red light
u-boot startup phase	off	on
Kernel boot to enter the system	on	off

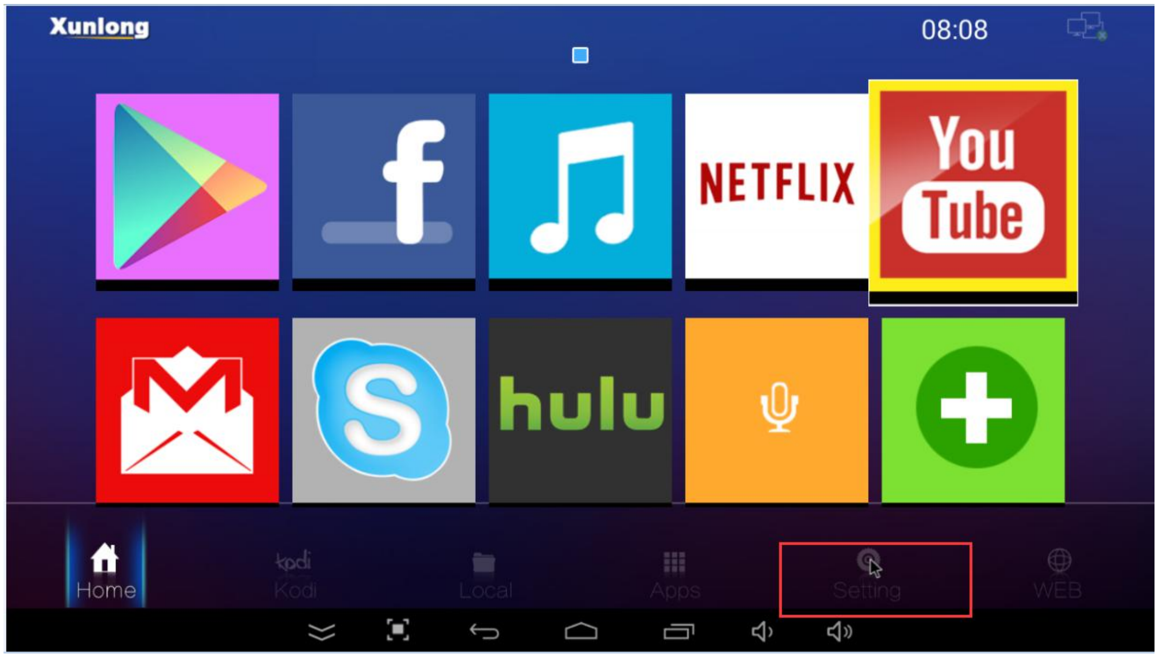
2) GPIO port corresponding to LED light

	GPIO
Green light	PL10
Red light	PA15

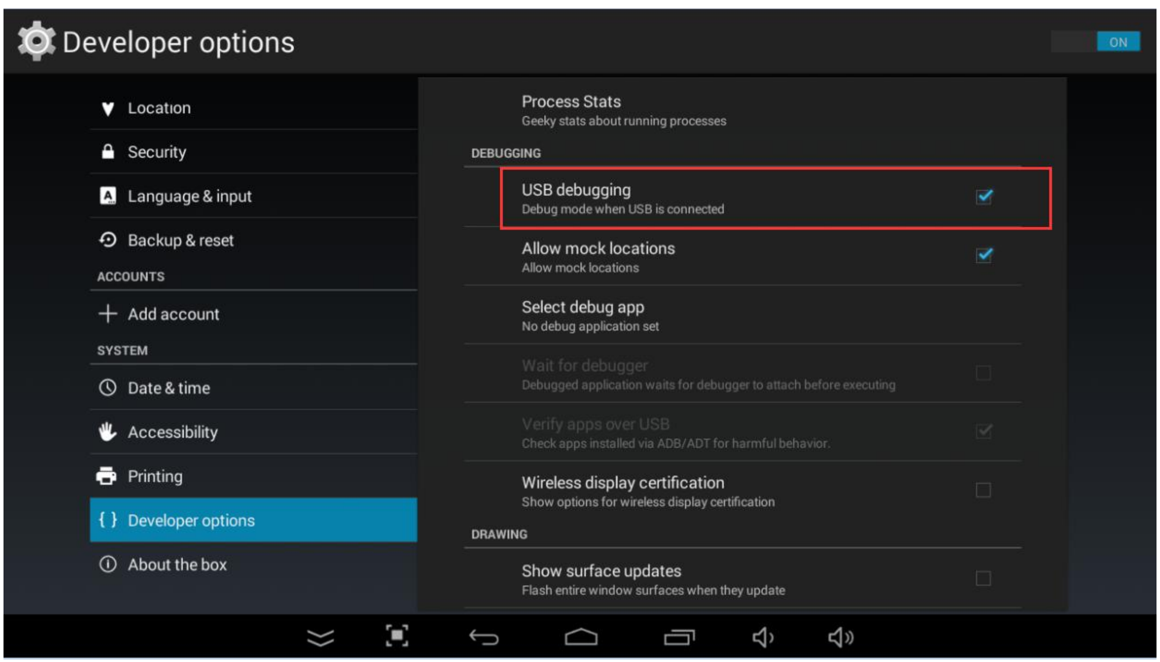
4. 5. How to use ADB

4. 5. 1. Android4.4 method to open the USB debugging option

1) Choose settings



2) Then find the developer option and make sure that USB debugging is turned on

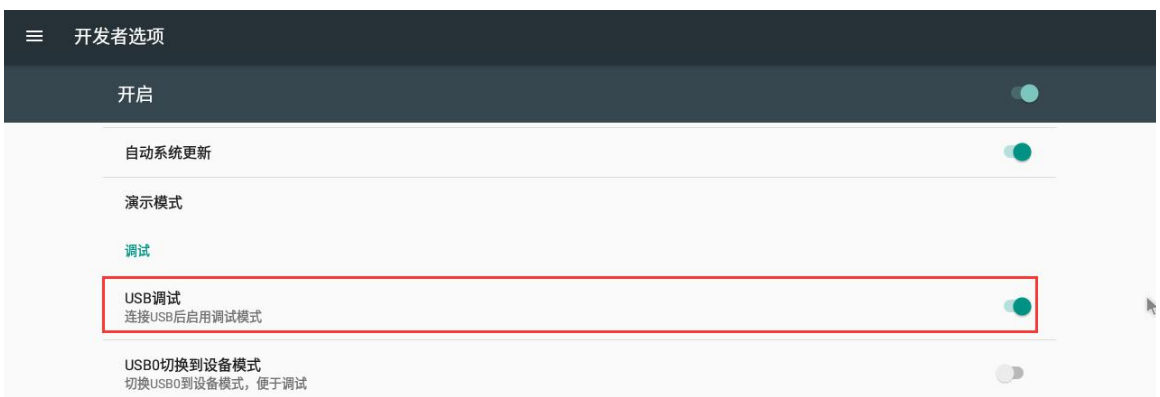


4. 5. 2. How to enable the USB debugging option in Android7.0

1) Choose settings

2) Then open the developer options

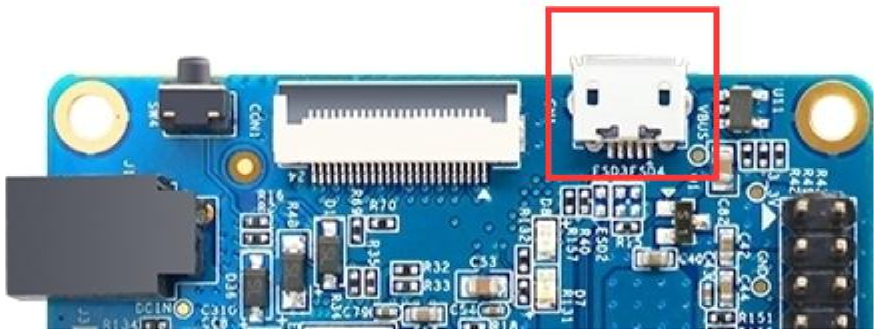
3) Then make sure that USB debugging is turned on



4. 5. 3. Use data cable to connect adb for debugging

1) First make sure that the **USB debugging option has been turned on**

2) Prepare a USB-to-Micro USB cable, insert the USB interface into the USB interface of the computer, and insert one end of the Micro USB interface into the USB OTG interface of the development board



3) Install adb tool on Ubuntu PC

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install adb
```

4) Then check if the adb device can be recognized

```
test@test:~$ adb devices
List of devices attached
20080411      device
```

5) Then you can log in to the android system through the adb shell on the Ubuntu PC

```
test@test:~$ adb shell
root@dolphin-fvd-p1:/ #
```

4. 5. 4. Use network connection adb debugging

1) The use of network adb does not require a USB to microphones USB cable to connect the computer and the development board, but communicates through the network, so first make sure that the network of the development board is connected

2) Then turn on the **USB debugging option**

3) Make sure that the **service.adb.tcp.port** of the Android system is set to port number 5555

```
root@dolphin-fvd-p1:/ # getprop | grep "adb.tcp"
[service.adb.tcp.port]: [5555]
```

3) If **service.adb.tcp.port** is not set, you can use the following command to set the port number of the network adb

```
root@dolphin-fvd-p1:/ # setprop service.adb.tcp.port 5555
root@dolphin-fvd-p1:/ # stop adbd
root@dolphin-fvd-p1:/ # start adbd
```

4) Install adb tool on Ubuntu PC

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install adb
```

5) Then connect to the network adb on the Ubuntu PC

```
test@test:~$ adb connect 192.168.1.xxx      (The IP address needs to be modified to the IP address of the development board)
connected to 192.168.1.149:5555
test@test:~$ adb devices
List of devices attached
192.168.1.xxx:5555          device
```

6) Finally, you can log in to the android system through the adb shell on the Ubuntu PC

```
test@test:~$ adb shell
root@dolphin-fvd-p1:/ #
```

4. 6. How to use USB camera

1) Insert the USB camera into the USB interface of the development board to ensure that the device node of the usb camera can be seen under /sys/class/videov4linux

```
root@dolphin-fvd-p1:/ # ls /sys/class/video4linux -l
lrwxrwxrwx root     root              2020-10-16 10:04 video0 -> ../../devices/platform/sunxi vfe.0/video4linux/video0
lrwxrwxrwx root     root              2020-10-16 10:04 video1 -> ../../devices/platform/sunxi-ehci.4/usb4/4-1/4-1:1.0/video4linux/video1
```

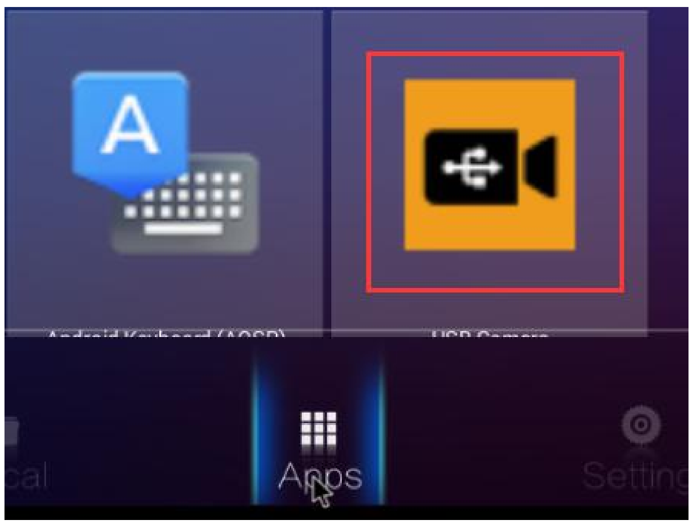
2) Download the USB camera test APP in Baidu cloud disk

3) Then install usbcamera.apk to the Android system, you can use U disk copy and install, you can also use adb to install, use adb to install usbcamera.apk command is

```
test@test:~$ adb devices
List of devices attached
192.168.1.xxx:5555   device      //First make sure that adb is properly connected

test@test:~$ adb install usbcamera.apk
```

4) After installation, you can see the startup icon of the USB camera in all applications



5) Then open the USB camera APP and you can see the video output of the USB camera

4. 7. How to use CSI camera

4. 7. 1. CSI camera interface specifications

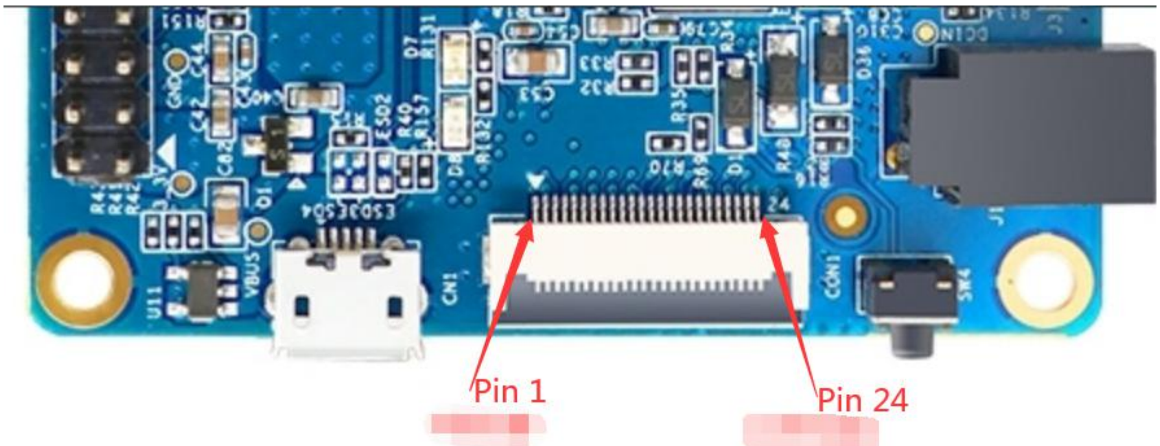
1) The CSI interface of Orange Pi PC supports two cameras, gc2035 and ov5640. The support for cameras in different systems is explained as follows

	GC2035	OV5640
Android 4.4	Support	Support
Android 7.0	No support	No support

2) The serial number of the CSI interface pins is shown in the figure below

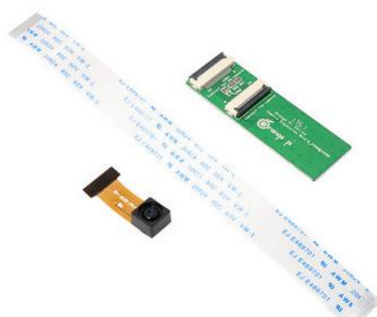
   a. The No. 1 pin of the CSI interface is connected to the No. 24 pin of the camera adapter board

   b. The 24th pin of the CSI interface is connected to the 1st pin of the camera adapter board

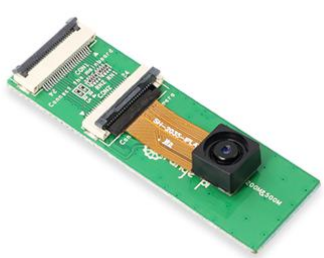


4. 7. 2. How to use gc2035 camera in Android4.4 system

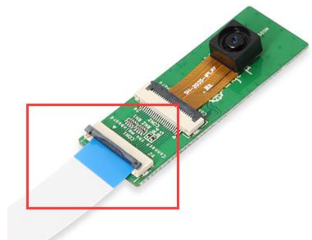
1) The Gc2035 camera kit includes a gc2035 camera, an adapter board and a cable



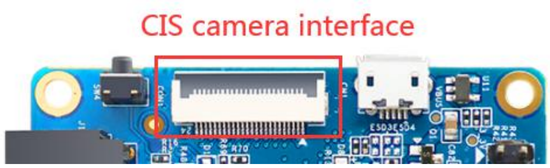
2) First insert the gc2035 camera into the adapter board

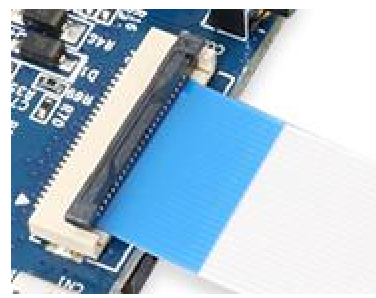


3) Then insert the ribbon cable into another card slot of the adapter board

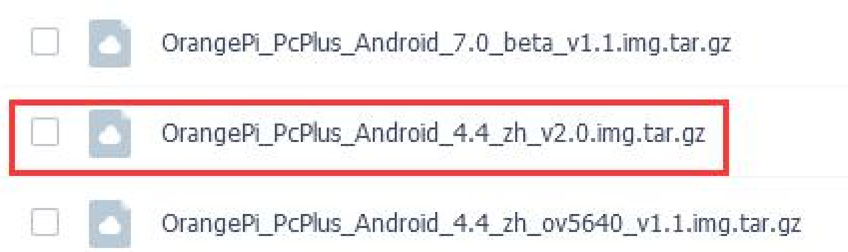


4) Then insert the other end of the cable into the CIS camera interface of the development board. Start the Android system after connecting the camera **(do not insert the camera after power-on)**

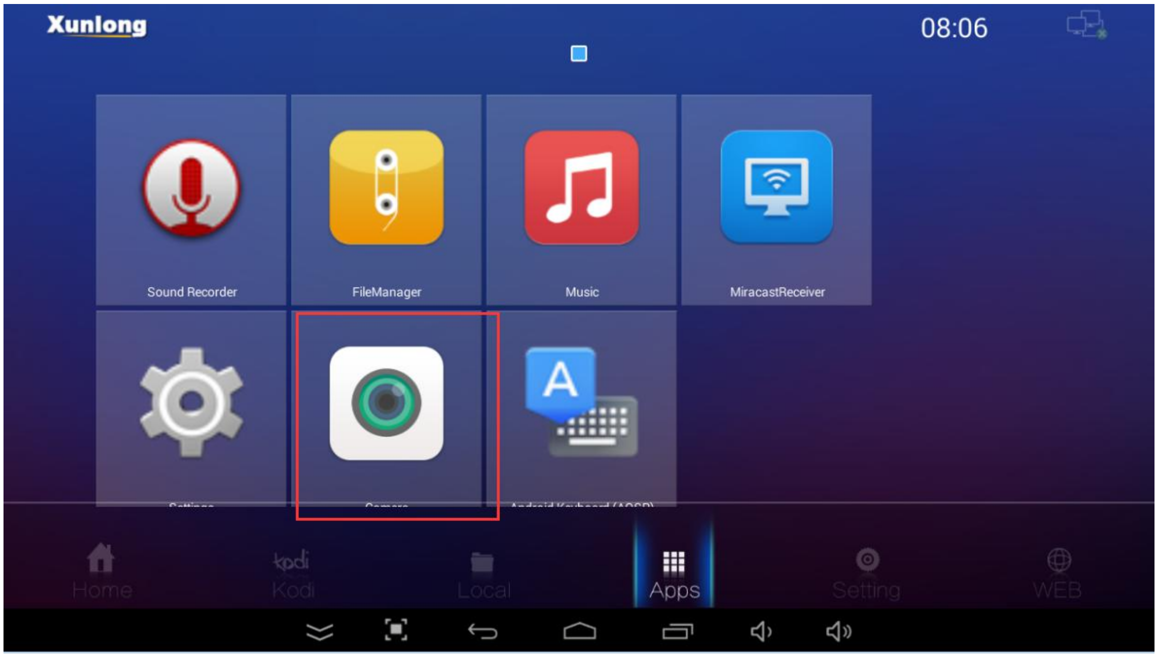




5) Android 4.4 system test gc2035 camera requires the following Android image

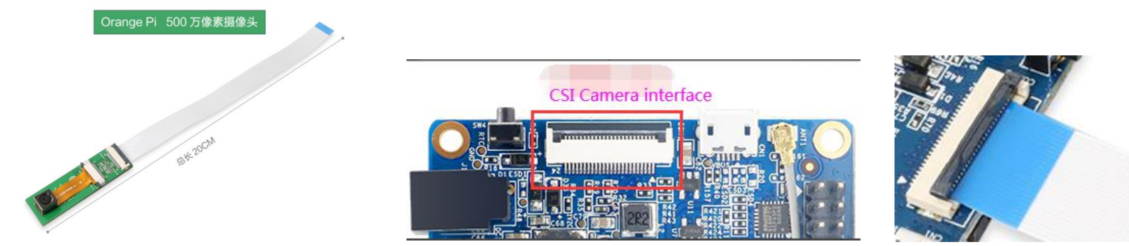


6) After the Android system is started, open the camera APP and you can see the output of the gc2035 camera. The location of the camera APP is shown in the figure below

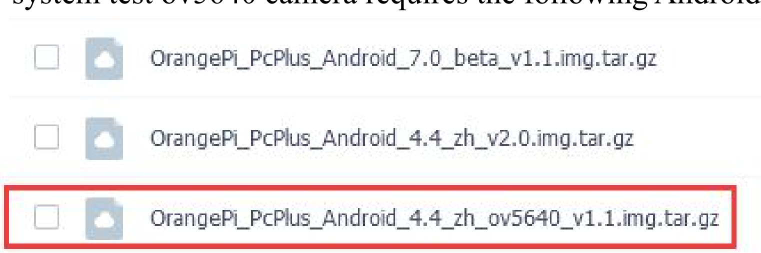


4. 7. 3. How to use the ov5640 camera in Android4.4 system

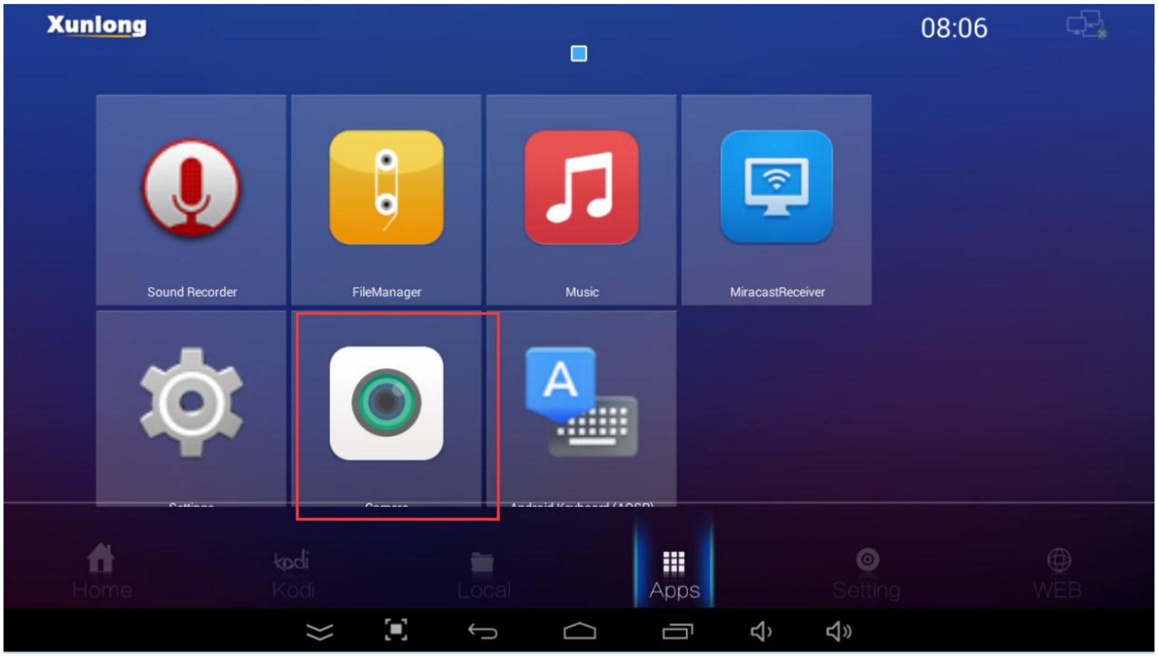
1) First connect the Ov5640 camera adapter board to the CIS camera interface of the development board through a cable, and then start the Android system after connecting the camera **(don't plug in the camera after powering on)**



2) Android 4.4 system test ov5640 camera requires the following Android image



3) After the Android system is started, open the camera APP and you can see the output of the ov5640 camera. The location of the camera APP is shown in the figure below



5. Linux SDK instructions

Linux SDK The compilation of the Linux SDK is performed on a **PC or virtual machine (VirtualBox or VMware)** with Ubuntu 18.04 installed. Please do not use other versions of the Ubuntu system or compile the Linux SDK on WSL

5. 1. Get the source code of linux sdk

5. 1. 1. Download orangepi-build from github

1) First download the code of orangepi-build. The code of orangepi-build is modified based on the armbian build system. At present, the H3 series development boards already support the legacy branch and the current branch.

```
test@test:~$ sudo apt update
test@test:~$ sudo apt install git
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git
```

To download the code of orangepi-build through the git clone command, you do not need to enter the username and password of the github account (the other codes in this manual are also the same), if you enter the git clone command, the Ubuntu PC prompts the user who needs to enter the github account Name and password, usually the address of the orangepi-build warehouse behind git clone is entered incorrectly. First of all, please check the spelling of the command carefully, instead of thinking that we forgot to provide the username and password of the github account.

2) The legacy branch uses the BSP version of the kernel. The current branch generally uses the u-boot and kernel close to the mainline version. The u-boot and linux kernel currently used by the H3 series development boards are as follows

Branch	u-boot version	linux kernel version
legacy	u-boot 2018.05	linux3.4.113
current	u-boot 2020.04	linux5.4.65

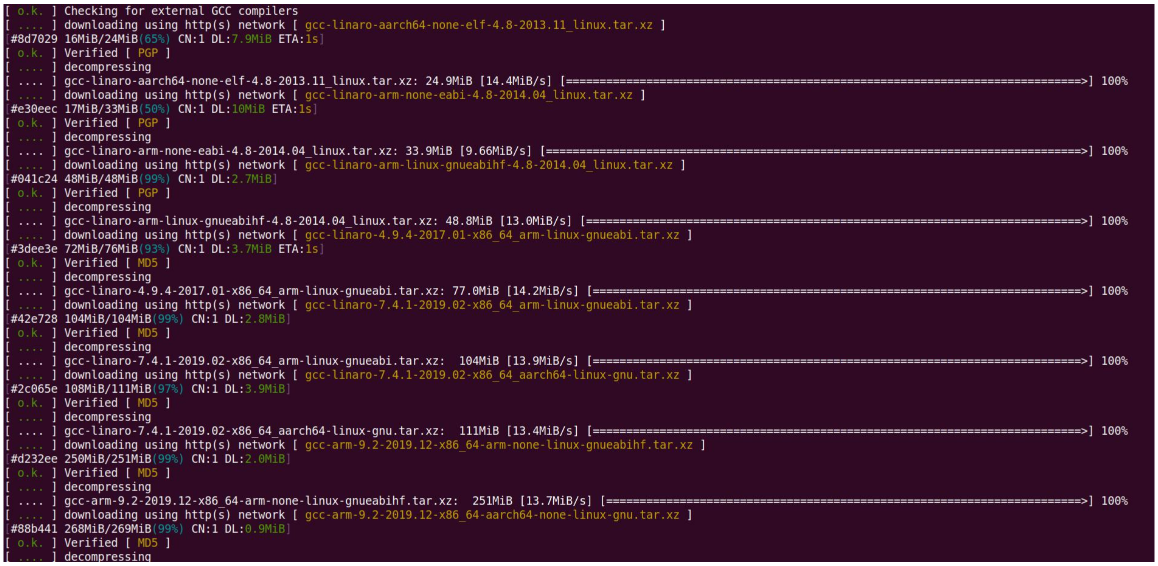
3) After orangepi-build is downloaded, it will contain the following files and folders

a. **build.sh**: Compile the startup script
b. **external**: Contains the configuration files needed to compile the image, specific scripts, and the source code of some programs, etc.
c. **LICENSE**: GPL 2 license file
d. **README.md**: d.orangepi-build documentation
e. **scripts**: General script for compiling linux image

```
test@test:~/orangepi-build$ ls
build.sh    external    LICENSE    README.md    scripts
```

5.1.2. Download the cross-compilation toolchain

1) When orangepi-build is run for the first time, it will automatically download the cross-compilation toolchain and place it in the toolchains folder. Every time the orangepi-build build.sh script is run, it will check whether the cross-compilation toolchain in toolchains exists. , If it does not exist, it will restart the download, if it exists, it will be used directly, and the download will not be repeated



2) The image URL of the cross-compilation tool chain in China is the open source software mirror site of Tsinghua University

```
https://mirrors.tuna.tsinghua.edu.cn/armbian-releases/_toolchain/
```

3) After **Toolchains** is downloaded, it will contain multiple versions of cross-compilation toolchains

```
test@test:~/orangepi-build$ ls toolchains/
gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
gcc-linaro-4.9.4-2017.01-x86_64_arm-linux-gnueabi
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
gcc-linaro-7.4.1-2019.02-x86_64_aarch64-linux-gnu
gcc-linaro-7.4.1-2019.02-x86_64_arm-linux-gnueabi
gcc-linaro-aarch64-none-elf-4.8-2013.11_linux
gcc-linaro-arm-linux-gnueabihf-4.8-2014.04_linux
gcc-linaro-arm-none-eabi-4.8-2014.04_linux
```

4) The cross-compilation tool chain used to compile the H3 linux kernel source code is

a. linux3.4

```
gcc-linaro-5.5.0-2017.10-x86_64_arm-linux-gnueabihf
```

b. linux5.4

```
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
```

5) The cross-compilation tool chain used to compile the H3 u-boot source code is

a. u-boot 2018.05

```
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
```

b. u-boot 2020.04

```
gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf
```

5.1.3. Description of the complete directory structure of orangepi-build

1) After the orangepi-build repository is downloaded, it does not contain the linux kernel, u-boot source code and cross-compilation tool chain. The source code of the linux kernel and u-boot are stored in a separate git repository **(please do not download and use the kernel and u separately -boot source code to compile, unless you know how to use it)**

a. The git repository stored in the linux kernel source code is as follows

a) linux3.4

```
https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-3.4-sun8i
```

b) linux5.4

```
https://github.com/orangepi-xunlong/linux-orangepi/tree/orange-pi-5.4
```

b. The git repository where u-boot source code is stored is as follows

a) u-boot 2018.05

```
https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2018.05-sun8i-linux3.4
```

b) u-boot 2020.04

```
https://github.com/orangepi-xunlong/u-boot-orangepi/tree/v2020.04
```

2) When orangepi-build runs for the first time, it will download the cross-compilation tool chain, u-boot and linux kernel source code. After successfully compiling a linux image, the files and folders that can be seen in orangepi-build are:

a. **build.sh**: Compile the startup script

b. **external**: Contains the configuration files needed to compile the image, scripts for specific functions, and the source code of some programs. The rootfs compressed package cached during the compiling of the image is also stored in external

c. **kernel**: Store the source code of the Linux kernel. The folder named orange-pi-3.4-sun8i stores the kernel source code of the legacy branch of the H3 development board, and the folder named orange-pi-5.4 stores the current branch of the H3 development board. The kernel source code (if only the linux image of the legacy branch is compiled, then only the kernel source code of the legacy branch can be seen; if only the linux image of the current branch is compiled, then only the kernel source code of the current branch can be seen), the kernel Please do not modify the name of the source code folder manually. If the build system is modified, the kernel source code will be downloaded again when the system is running.

d. **LICENSE**: GPL 2 license file
e. **README.md**: orangepi-build documentation
f. **output**: Store the compiled u-boot, linux and other deb packages, compilation logs, and compiled images and other files
g. **scripts**: General script for compiling linux image
h. **toolchains**: Store the cross-compilation tool chain
i. **u-boot**: Store the source code of u-boot, the folder named v2018.05-sun8i-linux3.4 inside stores the u-boot source code of the legacy branch of the H3 development board, and the folder named v2020.04 inside stores the H3 development U-boot source code of the current branch of the board (if only the linux image of the legacy branch is compiled, then you can only see the u-boot source code of the legacy branch; if you only compile the linux image of the current branch, then you can only see the current Branch u-boot source code), please do not modify the name of the u-boot source code folder manually. If the compilation system is modified, the u-boot source code will be re-downloaded when the system is running.
j. **userpatches**: Store configuration files needed to compile scripts

```
test@test:~/orangepi-build$ ls
build.sh    external    kernel    LICENSE    output    README.md    scripts
toolchains    u-boot    userpatches
```

5. 1. 4. Download from Google Cloud

```
Link:
http://www.orangepi.org/downloadresources/orangepipc/2019-11-12/oragepipc_14d99caf
ecec5b2d240caf09f1.html
```

5. 2. Compile u-boot

1) Run the build.sh script, remember to add sudo permissions

test@test:~/orangepi-build$ **sudo ./build.sh**

2) Select **U-boot package,** then press Enter

```
Choose an option
Compile image | rootfs | kernel | u-boot

            U-boot package
            Kernel package
            Rootfs and all deb packages
            Full OS image for flashing
```

3) Then select the model of the development board

```
Choose an option
Please choose a Board.

   orangepir1         Allwinner H2+ quad core 256MB RAM WiFi SPI 2xETH
   orangepizero       Allwinner H2+ quad core 256MB/512MB RAM WiFi SPI
   orangepipc         Allwinner H3 quad core 1GB RAM
   orangepipcplus     Allwinner H3 quad core 1GB RAM WiFi eMMC
   orangepione        Allwinner H3 quad core 512MB RAM
```

4) Then select the branch

a. current will compile u-boot v2020.04

b. legacy will compile u-boot v2018.05

```
Choose an option
Select the target kernel branch

          current Recommended. Come with best support
          legacy  Old stable / Legacy
```

5) Then it will start to compile u-boot, some of the information prompted during compilation are explained as follows

a. u-boot source version

[o.k.] Compiling u-boot [**v2020.04**]

b. The version of the cross-compilation toolchain

```
[ o.k. ] Compiler version [ arm-none-linux-gnueabihf-gcc 9.2.1 ]
```

c. Compile the generated u-boot deb package path

```
[ o.k. ] Target directory [ output/debs/u-boot ]
```

d. The package name of the compiled u-boot deb package

```
[ o.k. ] File name [ linux-u-boot-current-orangepipc_2.1.0_armhf.deb ]
```

e. Compile time

```
[ o.k. ] Runtime [ 1 min ]
```

f. Repeat the command to compile u-boot, use the following command without selecting through the graphical interface, you can directly start compiling u-boot

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh   BOARD=orangepipc
BRANCH=current BUILD_OPT=u-boot KERNEL_CONFIGURE=yes ]
```

6) View the compiled u-boot deb package

```
test@test:~/orangepi-build$ ls output/debs/u-boot/
linux-u-boot-current-orangepipc_2.1.0_armhf.deb
```

7) The files contained in the generated u-boot deb package are as follows

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ $ dpkg -x \
linux-u-boot-current-orangepipc_2.1.0_armhf.deb .
test@test:~/orangepi_build/output/debs/u-boot$ ls
linux-u-boot-current-orangepipc_2.1.0_armhf.deb    usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi_build/output/debs/u-boot$ tree usr
usr/
└── lib
    ├── linux-u-boot-current-orangepipc_2.1.0_armhf
    │   └── u-boot-sunxi-with-spl.bin     //u-boot 的二进制文件
    └── u-boot
```

```
        ├── LICENSE
        ├── orangepi_pc_plus_defconfig     //编译 u-boot 源码使用的配置文件
        └── platform_install.sh            //u-boot 的烧录脚本

3 directories, 4 files
```

8) When the orangepi-bulid compilation system compiles the u-boot source code, it will first synchronize the u-boot source code with the u-boot source code of the github server, so if you want to modify the u-boot source code, you first need to turn off the download and update function of the source code **(You need to compile u-boot once to turn off this function, otherwise you will be prompted that u-boot's source code cannot be found),** otherwise the changes made will be restored, the method is as follows:

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

9) When debugging u-boot code, you can use the following method to update u-boot in the linux image for testing

a. Upload the compiled u-boot deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs/u-boot
test@test:~/orangepi_build/output/debs/u-boot$ scp \
linux-u-boot-current-orangepipc_2.1.0_armhf.deb root@192.168.1.207:/root
```

b. Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt remove -y linux-u-boot-orangepipc-current
```

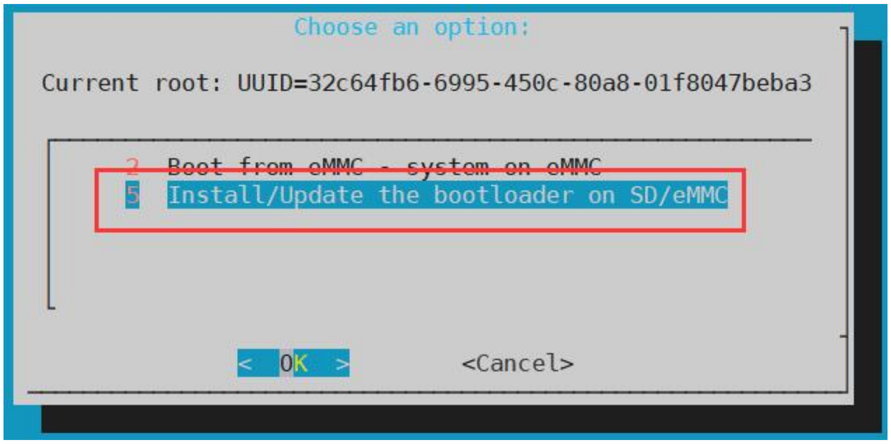
c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-u-boot-current-orangepipc_2.1.0_armhf.deb
```

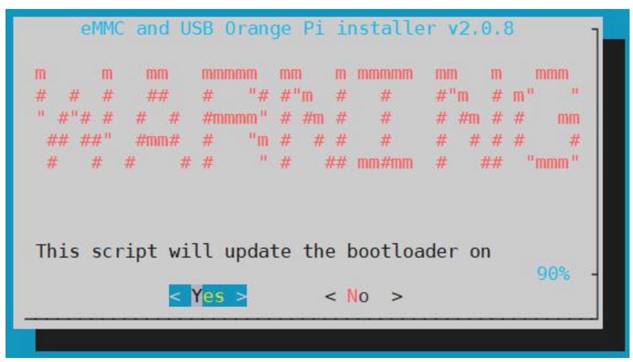
d. Then run the nand-sata-install script

```
root@orangepi:~# nand-sata-install
```

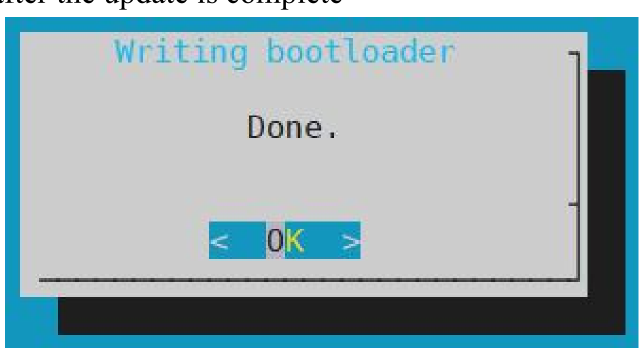
e. Then choose **5 Install/Update the bootloader on SD/eMMC**



f. After pressing the enter key, a Warring will pop up first



g. Press Enter again to start updating u-boot, and the following information will be displayed after the update is complete



Then you can restart to test that the u-boot modification is effective

5. 3. Compile the linux kernel

1) Run the build.sh script, remember to add sudo permissions

test@test:~/orangepi-build$ **sudo ./build.sh**

2) Select **Kernel package**, and then press Enter

```
Choose an option
Compile image | rootfs | kernel | u-boot

U-boot package
Kernel package
Rootfs and all deb packages
Full OS image for flashing
```

3) Then select the model of the development board

```
Choose an option
Please choose a Board.

orangepir1        Allwinner H2+ quad core 256MB RAM WiFi SPI 2xETH
orangepizero      Allwinner H2+ quad core 256MB/512MB RAM WiFi SPI
orangepipc        Allwinner H3 quad core 1GB RAM
orangepipcplus    Allwinner H3 quad core 1GB RAM WiFi eMMC
orangepione       Allwinner H3 quad core 512MB RAM
```

4) Then select the branch

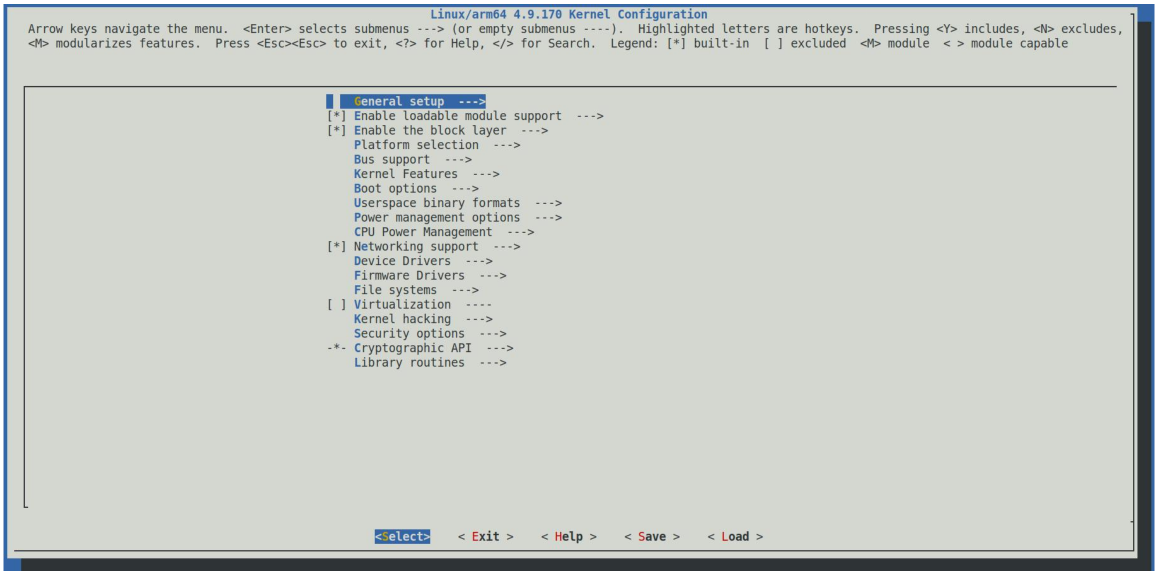
a. current will compile linux5.4

b. legacy will compile linux3.4

```
Choose an option
Select the target kernel branch

current Recommended. Come with best support
legacy  Old stable / Legacy
```

5) Then the kernel configuration interface opened through **make menuconfig** will pop up. At this time, you can directly modify the kernel configuration. If you don't need to modify the kernel configuration, just exit directly. After exiting, the kernel source code will be compiled.

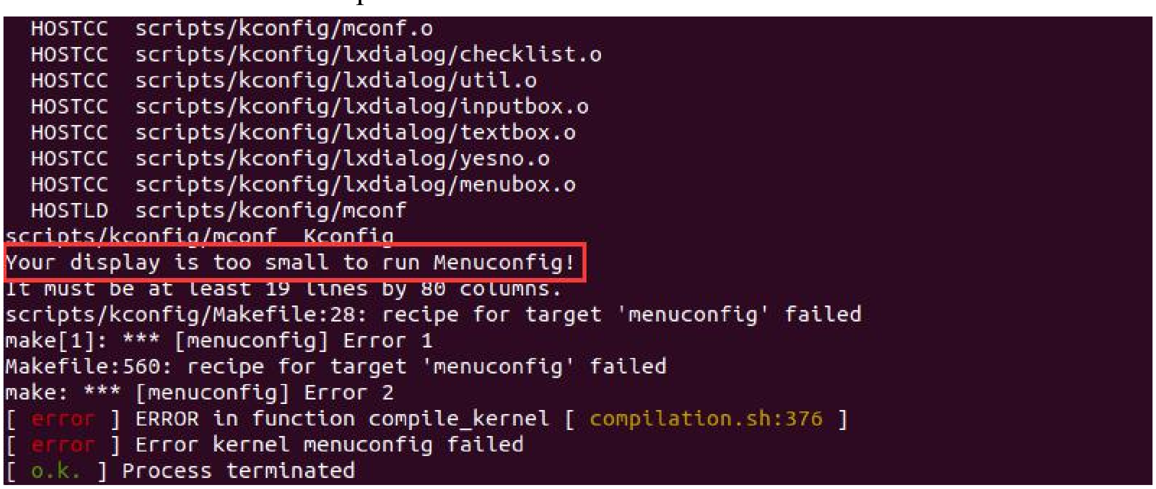


a. If you do not need to modify the configuration options of the kernel, when you run the build.sh script, pass in **KERNEL_CONFIGURE=no** to temporarily block the pop-up kernel configuration interface.

```
test@test:~/orangepi-build$ sudo ./build.sh KERNEL_CONFIGURE=no
```

b. You can also set **KERNEL_CONFIGURE=no** in the orangepi-build/userpatches/config-default.conf configuration file to disable this feature permanently

c. If the following error is prompted when compiling the kernel, this is because the terminal interface of the Ubuntu PC is too small, and the make menuconfig interface cannot be displayed. Please adjust the terminal of the Ubuntu PC to the maximum, and then re-run the build.sh script



6) When compiling the kernel source code, the following information will be prompted (take the current branch as an example)

a. The version of the kernel source code

```
[ o.k. ] Compiling legacy kernel [ 5.4.65 ]
```

b. The version of the cross-compilation tool chain used to compile the kernel source code

```
[ o.k. ] Compiler version [ aarch64-none-linux-gnu-gcc 9.2.1 ]
```

h. The configuration file used by the kernel by default and the path where it is stored

```
[ o.k. ] Using kernel config file [ config/kernel/linux-sunxi-current.config ]
```

i. The final configuration file .config used by the kernel (modified the default kernel configuration file through make menuconfig) will be copied to **output/config.** If the kernel configuration is not modified, the final configuration file and the default configuration file are Consistent

```
[ o.k. ] Exporting new kernel config [ output/config/linux-sunxi-current.config ]
```

j. The path of the deb package related to the kernel generated by the compilation

```
[ o.k. ] Target directory [ output/debs/ ]
```

k. The package name of the deb package containing the kernel image and kernel module generated by the compilation

```
[ o.k. ] File name [ linux-image-current-sunxi_2.1.0_armhf.deb ]
```

l. Compile time

```
[ o.k. ] Runtime [ 4 min ]
```

m. At the end, it will display the compiling command to recompile the kernel selected last time. Use the following command without selecting through the graphical interface, you can directly start compiling the kernel source code

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh    BOARD=orangepipc
BRANCH=current BUILD_OPT=u-boot KERNEL_CONFIGURE=yes ]
```

10) View the deb package related to the kernel image generated by the compilation

a. **linux-dtb-current-sunxi_2.1.0_armhf.deb** contains dtb files used by the kernel

b. **linux-headers-current-sunxi_2.1.0_armhf.deb** contains the header files used by the kernel

c. **linux-image-current-sunxi_2.1.0_armhf.deb** contains kernel images and kernel modules

```
test@test:~/orangepi-build$ ls output/debs/linux-*
output/debs/linux-dtb-current-sunxi_2.1.0_armhf.deb
output/debs/linux-image-current-sunxi_2.1.0_armhf.deb
output/debs/linux-headers-current-sunxi_2.1.0_armhf.deb
```

11) The files contained in the generated linux-image deb package are as follows

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-image-current-sunxi_2.1.0_armhf.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-image-current-sunxi_2.1.0_armhf.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot    etc    lib    linux-image-current-sunxi_2.1.0_armhf.deb    usr
```

b. The decompressed file is as follows

```
test@test:~/orangepi_build/output/debs/test$ tree -L 2
.
├── boot
│   ├── config-5.4.65-sunxi          //Configuration file used to compile the kernel
source code
│   ├── System.map-5.4.65-sunxi
│   └── vmlinuz-5.4.65-sunxi         //Compile the generated kernel image file
├── etc
│   └── kernel
├── lib
│   └── modules                      //Compile the generated kernel module
```

```
├── linux-image-current-sunxi_2.1.0_armhf.deb
└── usr
    ├── lib
    └── share

8 directories, 4 files
```

12) The files contained in the generated linux-dtb deb package are as follows

a. Use the following command to unzip the deb package

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ mkdir test
test@test:~/orangepi_build/output/debs$ cp \
linux-dtb-current-sunxi_2.1.0_armhf.deb test/
test@test:~/orangepi_build/output/debs$ cd test
test@test:~/orangepi_build/output/debs/test$ dpkg -x \
linux-dtb-current-sunxi_2.1.0_armhf.deb .
test@test:~/orangepi_build/output/debs/test$ ls
boot   linux-image-current-sunxi_2.1.0_armhf.deb   usr
```

b. Use the following command to unzip the deb package

```
test@test:~/orangepi_build/output/debs/test$ tree -L 2
.
├── boot
│   └── dtb-5.4.65-sunxi              //Store dtb files used by the kernel
├── linux-dtb-current-sunxi_2.1.0_armhf.deb
└── usr
    └── share

4 directories, 1 file
```

13) When the orangepi-bulid compilation system compiles the linux kernel source code, it first synchronizes the linux kernel source code with the linux kernel source code of the github server, so if you want to modify the linux kernel source code, you need to turn off the source code update function first **(you need to compile the linux kernel once. This function can only be turned off after the source code, otherwise it will be prompted**

that the source code of the linux kernel cannot be found), otherwise the changes made will be restored, the method is as follows:

Set the IGNORE_UPDATES variable in userpatches/config-default.conf to "yes"

```
test@test:~/orangepi-build$ vim userpatches/config-default.conf
IGNORE_UPDATES="yes"
```

14) If you modify the kernel, you can use the following method to update the kernel and kernel modules of the Linux system on the development board

a. Upload the compiled linux deb package to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi_build/output/debs$ scp \
linux-image-current-sunxi_2.1.0_armhf.deb root@192.168.1.207:/root
```

b. Then log in to the development board and uninstall the installed deb package of u-boot

```
root@orangepi:~# apt purge -y linux-image-current-sunxi
```

c. Install the new u-boot deb package just uploaded

```
root@orangepi:~# dpkg -i linux-image-current-sunxi_2.1.0_armhf.deb
```

d. Then restart the development board, and then check whether the kernel-related changes have taken effect

15) The method of installing the kernel header file into the linux system is as follows

a. Upload the deb package of the compiled linux header file to the linux system of the development board

```
test@test:~/orangepi-build$ cd output/debs
test@test:~/orangepi-build/output/debs$ scp \
linux-headers-current-sunxi_2.1.0_armhf.deb root@192.168.1.207:/root
```

b. Then log in to the development board and install the deb package of the linux header file just uploaded

```
root@orangepi:~# dpkg -i linux-headers-current-sunxi_2.1.0_armhf.deb
```

c. After installation, you can see the contents of the kernel header file just installed in /usr/src

```
root@orangepi:~# ls /usr/src
linux-headers-current-sunxi
root@orangepi:~# ls /usr/src/linux-headers-current-sunxi
Documentation  Module.symvers  certs  firmware  init  lib  net  security  usr
Kconfig  arch  crypto  fs  ipc  mm  samples  sound  virt  Makefile  block
drivers  include  kernel  modules  scripts  tools
```

5. 4. Compile rootfs

1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select **Rootfs and all deb packages**, and then press Enter

```
Choose an option
Compile image | rootfs | kernel | u-boot

            U-boot package
            Kernel package
            Rootfs and all deb packages
            Full OS image for flashing
```

3) Then select the model of the development board

```
Choose an option
Please choose a Board.

   orangepir1        Allwinner H2+ quad core 256MB RAM WiFi SPI 2xETH
   orangepizero      Allwinner H2+ quad core 256MB/512MB RAM WiFi SPI
   orangepipc        Allwinner H3 quad core 1GB RAM
   orangepipcplus    Allwinner H3 quad core 1GB RAM WiFi eMMC
   orangepione       Allwinner H3 quad core 512MB RAM
```

4) Then select the type of rootfs

buster	Debian 10
bionic	Ubuntu 18.04
focal	Ubuntu 20.04
xenial	Ubuntu16.04

a. Linux distributions supported by linux5.4 are as follows

```
Choose an option
Select the target OS release package base

buster Debian 10 Buster
bionic Ubuntu Bionic 18.04 LTS
focal  Ubuntu Focal 20.04 LTS
```

b. The Linux distributions supported by linxu3.4 are as follows

```
Choose an option
Select the target OS release package base

xenial Ubuntu Xenial 16.04 LTS
```

5) Then select the type of image

a. **Image with console interface** represents the image of the server version, which is relatively small

b. **Image with desktop environment** means that the image of desktop version, and the volume is relatively large

```
Choose an option
Select the target image type.

Image with console interface (server)
Image with desktop environment
```

6) If it is to compile the image of the server version, you can also choose to compile the Standard version or the Minimal version. The pre-installed software of the Minimal version will be much less than the Standard version.

```
Choose an option
Select the target image type.

Standard image with console interface
Minimal image with console interface
```

7) After selecting the type of image, rootfs will be compiled, and the following information will be prompted during compilation

a. Type of rootfs

[o.k.] local not found [Creating new rootfs cache for **bionic**]

b. The storage path of the compiled rootfs compressed package

[o.k.] Target directory [**external/cache/rootfs**]

a. The name of the rootfs compressed package generated by the compilation

```
[ o.k. ] File name [ bionic-cli-armhf.153618961f14c28107ca023429aa0eb9.tar.lz4 ]
```

b. Compilation time

```
[ o.k. ] Runtime [ 13 min ]
```

c.

d. Repeat the command to compile rootfs, use the following command without selecting through the graphical interface, you can start compiling rootfs directly

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh BOARD=orangepipc BRANCH=current BUILD_OPT=rootfs RELEASE=bionic BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

8) View the compiled rootfs compressed package

a. bionic-cli-armhf.153618961f14c28107ca023429aa0eb9.tar.lz4 is a compressed package of rootfs, the meaning of each field of the name is

a) **Bionic** represents the type of linux distribution of rootfs

b) **Cli** indicates that rootfs is the server version type, if it is dekstop, it indicates the desktop version type

c) **Armhf** indicates the architecture type of rootfs

d) **153618961f14c28107ca023429aa0eb9** is the MD5 hash value generated by the package names of all software packages installed by rootfs. As long as the list of software packages installed by rootfs is not modified, this value will not change. The compilation script will judge by this MD5 hash value Do you need to recompile rootfs

b. bionic-cli-armhf.153618961f14c28107ca023429aa0eb9.tar.lz4.list lists the package names of all packages installed by rootfs

```
test@test:~/orangepi_build$ ls external/cache/rootfs/
bionic-cli-armhf.153618961f14c28107ca023429aa0eb9.tar.lz4
bionic-cli-armhf.153618961f14c28107ca023429aa0eb9.tar.lz4.list
```

9) If the required rootfs already exists under **external/cache/rootfs**, then compiling the rootfs again will skip the compilation process and will not restart the compilation. When compiling the image, it will also go to **external/cache/rootfs** to find out whether it is already Rootfs with cache available, if available, use it directly, which can save a lot of downloading and compiling time

10) Since it takes a long time to compile rootfs, if you don't want to compile rootfs from scratch, or if there is a problem with compiling rootfs, you can directly download the rootfs compressed package cached by Orange Pi. The download link of rootfs compressed package Baidu cloud disk is shown below, download A good rootfs compressed package (don't decompress it) needs to be placed in the **external/cache/rootfs** directory of orangepi-build before it can be used normally by the compiled script

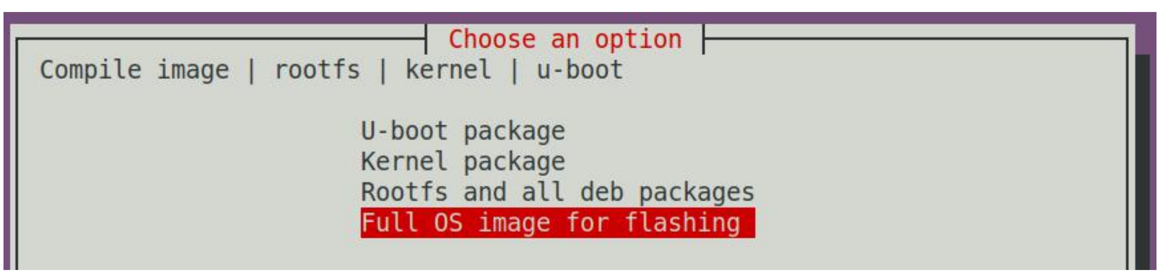
```
Link：https://pan.baidu.com/s/1vWQmCmSYdH7iCDFyKpJtVw
Code：zero
```

5. 5. Compile linux image

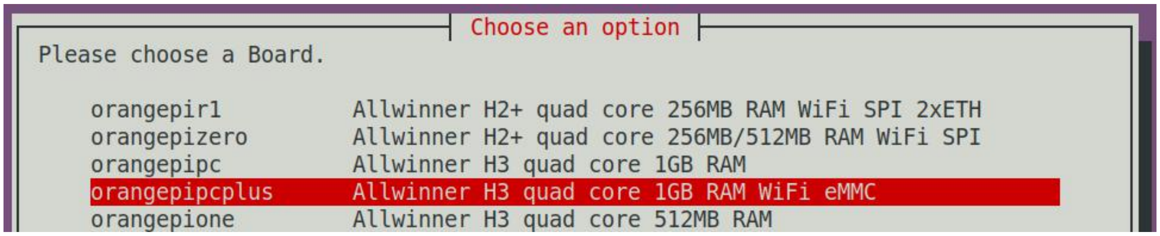
1) Run the build.sh script, remember to add sudo permissions

```
test@test:~/orangepi-build$ sudo ./build.sh
```

2) Select Full OS image for flashing, and then press Enter



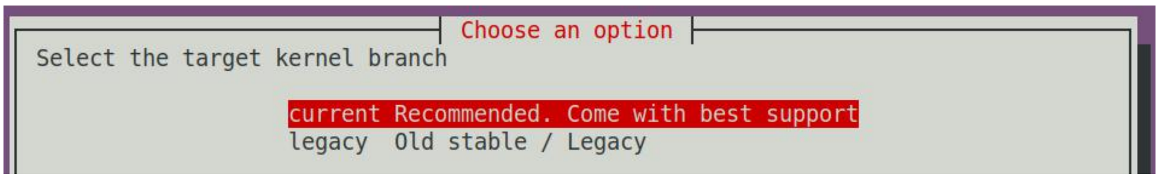
3) Then select the model of the development board



4) Then select the branch

a. Current will compile u-boot v2020.04、linux5.4

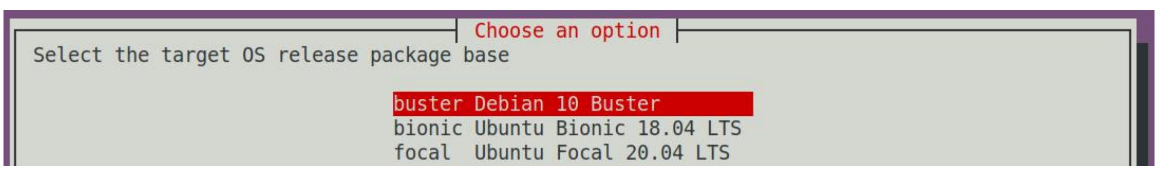
b. legacy will compile u-boot v2018.05、linux3.4



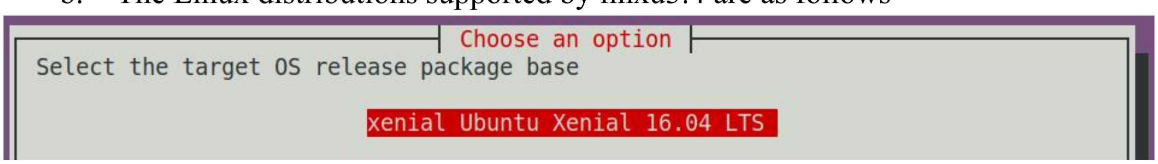
5) Then select the type of rootfs

buster	Debian 10
bionic	Ubuntu 18.04
focal	Ubuntu 20.04
xenial	Ubuntu16.04

a. Linux distributions supported by linux5.4 are as follows



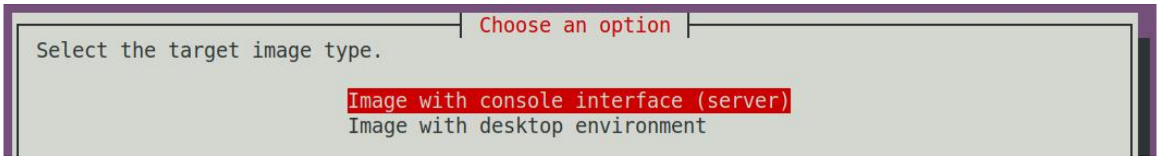
b. The Linux distributions supported by linxu3.4 are as follows



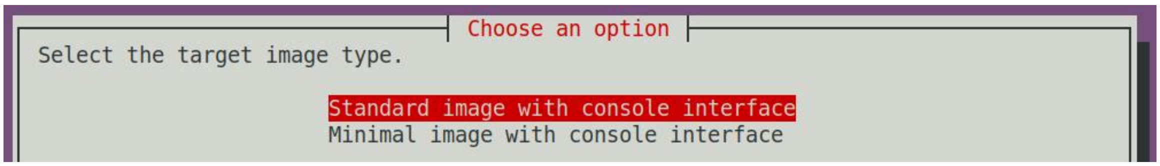
6) Then select the type of image

a. **Image with console interface** represents the image of the server version, which is relatively small

b. **Image with desktop environment** indicates that the image of desktop version and the volume is relatively large



7) If it is to compile the image of the server version, you can also choose to compile the Standard version or the Minimal version. The pre-installed software of the Minimal version will be much less than the Standard version.



8) After selecting the type of image, it will start to compile the Linux image. The general process of compiling the image is as follows

a. Compile u-boot source code and generate u-boot deb package
b. Compile linux source code, generate linux related deb package
c. Make deb package of linux firmware
d. Make deb package of orangepi-config tool
e. Make board-level support deb package
f. If it is to compile the desktop version image, the desktop related deb package will also be made
g. Check whether the rootfs has been cached, if there is no cache, re-create the rootfs, if it has been cached, just unzip and use
h. Install the previously generated deb package into rootfs
i. Make some specific settings for different development boards and different types of images, such as pre-installing additional software packages and modifying configuration files
j. Then make an image file and format the partition, the default type is ext4
k. Then copy the configured rootfs to the image partition
l. Then update the initramfs
m. Finally, write the bin file of u-boot to the image through the dd command

9) After compiling the image, the following information will be prompted

a. The storage path of the compiled linux image

```
[ o.k. ] Done building
[ output/images/orangepipc_2.1.0_ubuntu_bionic_server_linux5.4.65/orangepipc_2.1.0_ubuntu_bionic_server_linux5.4.65.img ]
```

b. The time used to compile the image

```
[ o.k. ] Runtime [ 9 min ]
```

c. Repeat the command to compile the image, use the following command without selecting through the graphical interface, you can directly start to compile the image

```
[ o.k. ] Repeat Build Options [ sudo ./build.sh BOARD=orangepipc BRANCH=current BUILD_OPT=image RELEASE=bionic BUILD_MINIMAL=no BUILD_DESKTOP=no KERNEL_CONFIGURE=yes ]
```

6. Android SDK instructions

1) The Android SDK supported by Allwinner H3 SOC is as follows

Android version	Kernel version
Android 4.4	linux 3.4
Android 7.0	linux 4.4

2) The compilation of the Android SDK is performed on a PC with **Ubuntu 14.04** installed, and there may be some differences in other versions of Ubuntu systems

3) Android 4.4 has more complete drivers than Android 7.0. Both versions of the SDK are the original SDK released by the chip manufacturer. If you want to use the Android images compiled by these SDKs on the Orange Pi development board, you need to target different boards. Adaptation can ensure the normal use of all functions

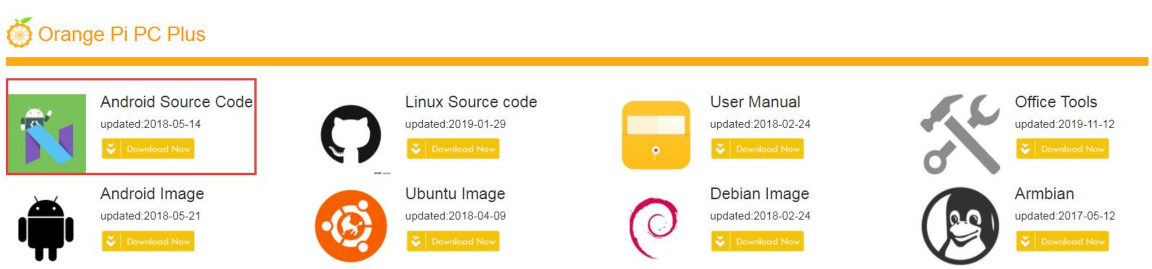
6. 1. Android 4.4 SDK instructions

6. 1. 1. Download the source code of android 4.4 sdk

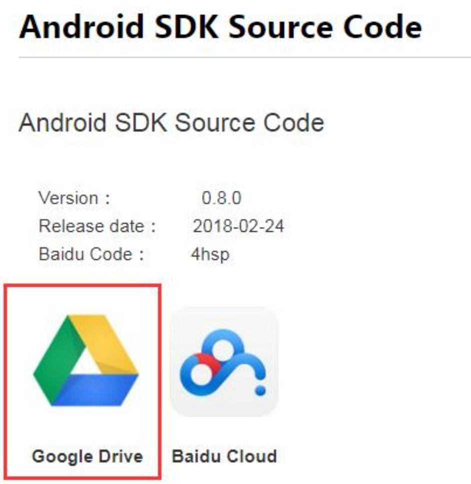
1) The download address of the Android source code is

http://www.orangepi.org/downloadresources/

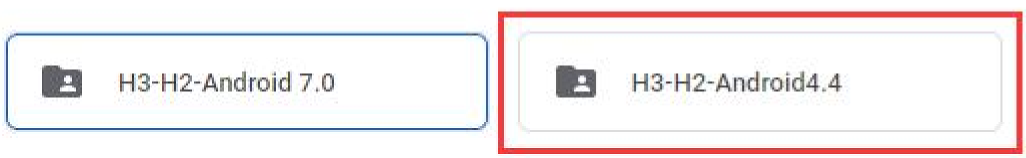
2) After entering the data page, find the data download link corresponding to the development board, and select the Android source code option



3) Then select Google Cloud



4) Then download the source code of H3-Android4.4



5) H3 android 4.4 source code contains the following 2 files

a. **OrangepiH3.tar**： android source code

b. **OrangepiH3.tar.md5sum**： The MD5 checksum file of OrangepiH3.tar

6) After downloading the android source code, first check whether the MD5 checksum is correct, if not, please download the source code again

```
test@test:~$ md5sum -c    OrangepiH3.tar.md5sum
OrangepiH3.tar: 确定
```

7) Then decompress the source code of android sdk, after decompressing the sdk, two folders of android and lichee will be generated

a. android: Store android-related code

b. linchee: Store the linux kernel and u-boot code

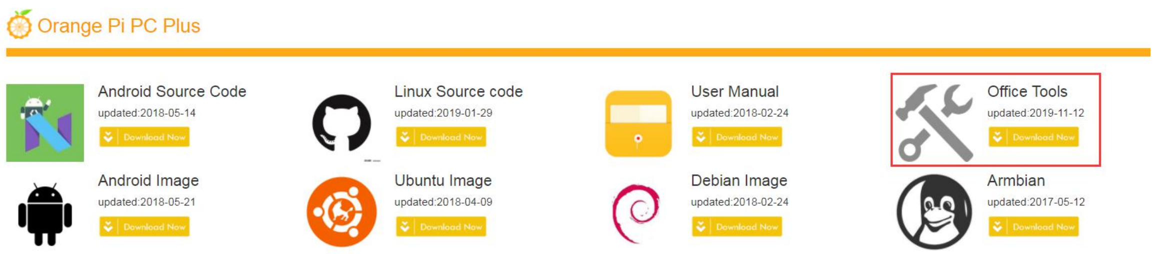
```
test@test:~$ mkdir OrangePiH3
test@test:~$ tar -xf OrangePiH3.tar -C OrangePiH3
```

```
test@test:~$ cd OrangePiH3
test@test:~/OrangePiH3$ ls
android    lichee
```

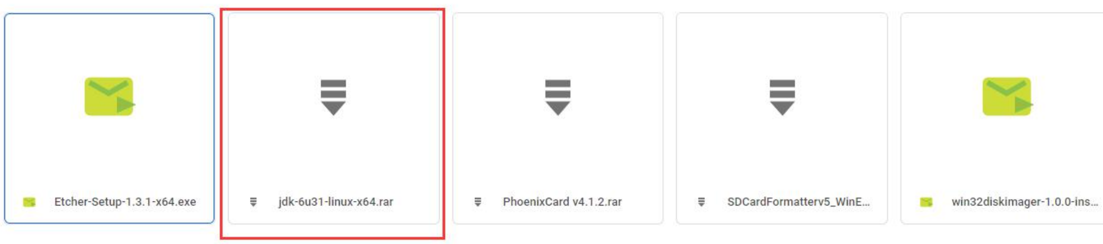
6. 1. 2. Build android compilation environment

1) Download the jdk installation package

a. Select the official tool on the data download page



b. Then select jdk-6u31-linux-x64.rar in the opened Baidu cloud disk



2) Install jdk, execute the following command, a folder named jdk1.6.0_31 will be generated under /usr/lib/jvm/ b. Then select jdk-6u31-linux-x64.rar in the opened Baidu cloud disk

```
test@test:~$ sudo cp jdk-6u31-linux-x64.bin /usr/lib/jvm/
test@test:~$ cd /usr/lib/jvm/
test@test:~ /usr/lib/jvm$ sudo chmod a+x ./jdk-6u31-linux-x64.bin
test@test:~ /usr/lib/jvm$ sudo ./jdk-6u31-linux-x64.bin
test@test:~ /usr/lib/jvm$ ls
jdk1.6.0_31   jdk-6u31-linux-x64.bin
```

3) Export java environment variables

```
test@test:~$ export JAVA_HOME=/usr/lib/jvm/jdk1.6.0_31
```

```
test@test:~$ export JRE_HOME=/usr/lib/jvm/jdk1.6.0_31/jre
test@test:~$ export CLASSPATH=.:$JAVA_HOME/lib:$JRE_HOME/lib:$CLASSPATH
test@test:~$ export PATH=$JAVA_HOME/bin:$JRE_HOME/bin:$JAVA_HOME:$PATH
```

4) Install platform support software

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install git gnupg flex bison gperf build-essential zip \
curl libc6-dev libncurses5-dev:i386 x11proto-core-dev \
libx11-dev:i386 libreadline6-dev:i386 libgl1-mesa-glx:i386 \
libgl1-mesa-dev g++-multilib mingw32 tofrodos \
python-markdown libxml2-utils xsltproc zlib1g-dev:i386
```

5) The location where the cross-compilation tool chain used in the compilation process is stored is

```
test@test:~$ cd OrangePiH3/lichee/brandy/gcc-linaro
test@test:~/OrangePiH3/lichee/brandy/gcc-linaro$ ls
arm-linux-gnueabi    bin    lib    libexec
```

6. 1. 3. Compile android image

6. 1. 3. 1. Compile the Linux kernel source code

1) The compilation environment needs to be configured when compiling the kernel for the first time. After the configuration, the kernel code will be compiled automatically

```
test@test:~/OrangePiH3$ cd lichee
test@test:~/OrangePiH3/lichee$ ./build.sh config

Welcome to mkscript setup progress
All available chips:
    0. sun8iw6p1
    1. sun8iw7p1
    2. sun8iw8p1
    3. sun9iw1p1
Choice: 1
All available platforms:
```

```
    0. android
    1. dragonboard
    2. linux
Choice: 0
All available business:
    0. dolphin
    1. secure
    2. karaok
Choice: 0
```

2) After compiling, the following information will be output

```
sun8iw7p1 compile Kernel successful


INFO: build kernel OK.
INFO: build rootfs ...
INFO: skip make rootfs for android
INFO: build rootfs OK.
----------------------------------------
build sun8iw7p1 android dolphin lichee OK
----------------------------------------
```

3) If you recompile the kernel code later, you only need to enter the **./build.sh** command to start compiling

```
test@test:~/OrangePiH3/lichee$ ./build.sh
```

6. 1. 3. 2. Compile android source code

1) The command to compile android is as follows

```
test@test:~$ cd android
test@test:~/OrangePiH3/android$ source build/envsetup.sh
test@test:~/OrangePiH3/android$ lunch dolphin_fvd_p2-eng
test@test:~/OrangePiH3/android$ extract-bsp
test@test:~/OrangePiH3/android$ make -j8 && pack
```

2) The final output log of the packaged and generated android image is as follows

```
test@test:~/OrangePiH3/android$ pack
......

----------image is at----------

lichee/tools/pack/sun8iw7p1_android_dolphin-p2_uart0.img

pack finish
```

3) The path where the generated Android image is stored is

```
lichee/tools/pack/sun8iw7p1_android_dolphin-p2_uart0.img
```

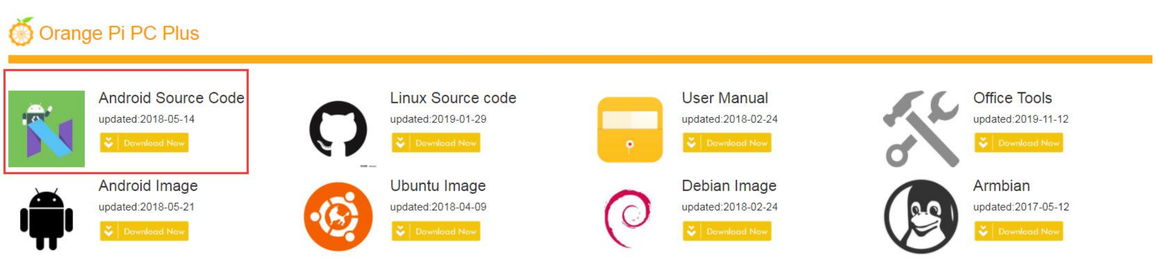
6. 2. Android 7.0 SDK instructions

6. 2. 1. Download the source code of android 7.0 sdk

1) The download address of the Android source code is

```
http://www.orangepi.org/downloadresources/
```

2) After entering the data page, find the data download link corresponding to the development board, and select the Android source code option



3) Then select Google Cloud

4) Then download the source code of H3-Android7.0

5) H3's android 7.0 source code includes file descriptions as follows

c. **H3-sdk7.0-2017-11-03.tar.gza*：** Sub-volume compressed package of android sdk source code

d. **md5sum.txt：** H3-sdk7.0-2017-11-03.tar.gza* of H3-sdk7.0-2017-11-03.tar.gza*

6) After downloading the android source code, first check whether the MD5 checksum is correct, if not, please download the source code again

```
test@test:~$ cd H3-Android7.0
test@test:~/H3-Android7.0$ md5sum -c md5sum.txt
H3-sdk7.0-2017-11-03.tar.gzaa: 确定
H3-sdk7.0-2017-11-03.tar.gzab: 确定
H3-sdk7.0-2017-11-03.tar.gzac: 确定
H3-sdk7.0-2017-11-03.tar.gzad: 确定
H3-sdk7.0-2017-11-03.tar.gzae: 确定
H3-sdk7.0-2017-11-03.tar.gzaf: 确定
H3-sdk7.0-2017-11-03.tar.gzag: 确定
```

```
H3-sdk7.0-2017-11-03.tar.gzah: 确定
H3-sdk7.0-2017-11-03.tar.gzai: 确定
H3-sdk7.0-2017-11-03.tar.gzaj: 确定
H3-sdk7.0-2017-11-03.tar.gzak: 确定
H3-sdk7.0-2017-11-03.tar.gzal: 确定
H3-sdk7.0-2017-11-03.tar.gzam: 确定
H3-sdk7.0-2017-11-03.tar.gzan: 确定
```

7) Then add multiple compressed packages and merge them into one compressed file

```
test@test:~/H3-Android7.0$ cat H3-sdk7.0-2017-11-03.tar.gza* > OrangePiH3.tar
```

8) Then decompress the source code of android sdk, after decompressing the sdk, two folders of android and lichee will be generated

a. android: Store android-related code

b. lichee: Store the linux kernel and u-boot code

```
test@test:~$ mkdir OrangePiH3
test@test:~$ tar -xf OrangePiH3.tar -C OrangePiH3
test@test:~$ cd OrangePiH3
test@test:~/OrangePiH3$ ls
android    lichee
```

6. 2. 2. Build android compilation environment

1) Install jdk

```
test@test:~$ sudo add-apt-repository ppa:openjdk-r/ppa
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install openjdk-8-jdk
```

2) Configure java environment variables

a. First determine the installation path of java, generally

```
test@test:~$ ls /usr/lib/jvm/java-8-openjdk-amd64
ASSEMBLY_EXCEPTION   bin   docs   include   jre   lib   man   src.zip
THIRD_PARTY_README
```

b. Then use the following command to export java environment variables

```
test@test:~$ export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

```
test@test:~$ export PATH=$JAVA_HOME/bin:$PATH
test@test:~$ export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

3) Install platform support software

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install git gnupg flex bison gperf build-essential \
zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 \
lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z1-dev ccache \
libgl1-mesa-dev libxml2-utils xsltproc unzip

test@test:~$ sudo apt-get install u-boot-tools
```

6. 2. 3. Compile android image

6. 2. 3. 1. Compile the kernel

1) First configure the compilation environment, after the configuration, the kernel source code will be compiled

```
test@test:~/H3-Android7.0 /OrangePiH3$ cd  lichee
test@test:~/H3-Android7.0 /OrangePiH3/lichee$ ./build.sh  config

./build.sh config

Welcome to mkscript setup progress
All available chips:
   0. sun50iw1p1
   1. sun50iw2p1
   2. sun50iw6p1
   3. sun8iw11p1
   4. sun8iw12p1
   5. sun8iw6p1
   6. sun8iw7p1
   7. sun8iw8p1
   8. sun9iw1p1
Choice: 6
```

```
All available platforms:
    0. android
    1. dragonboard
    2. linux
    3. camdroid
Choice: 0
All available business:
    0. dolphin
    1. secure
    2. karaok
Choice: 0
```

2) After compiling, the following information will be output

```
sun8iw7p1 compile Kernel successful



INFO: build kernel OK.
INFO: build rootfs ...
INFO: skip make rootfs for android
INFO: build rootfs OK.
----------------------------------------
build sun8iw7p1 android dolphin lichee OK
----------------------------------------
```

3) If you recompile the kernel code later, you only need to enter the **./build.sh** command to start compiling

```
test@test:~/OrangePiH3/lichee$ ./build.sh
```

6. 2. 3. 2. Compile android source code

1) The command to compile android is as follows

```
test@test:~/H3-Android7.0 /OrangePiH3$ cd android
test@test:~/H3-Android7.0 /OrangePiH3/android$ source build/envsetup.sh
test@test:~/H3-Android7.0 /OrangePiH3/android$ lunch dolphin_fvd_p1-eng
test@test:~/H3-Android7.0 /OrangePiH3/android$ extract-bsp
```

```
test@test:~/H3-Android7.0 /OrangePiH3/android$ make -j8 && pack
```

2) The final output log of the packaged and generated android image is as follows

```
----------image is at----------

lichee/tools/pack/sun8iw7p1_android_dolphin-p1_uart0.img

pack finish
```

4) The path where the generated Android image is stored is

```
lichee/tools/pack/sun8iw7p1_android_dolphin-p1_uart0.img
```